THEOREMS FOR FINITE AUTOMATA

by

Reverdy Edmond Wright

A Dissertation Presented to the Graduate Council of
The University of Florida
in Partial Fulfillment of the Requirements for the
Degree of Doctor of Philosophy

UNIVERSITY OF FLORIDA

1971

To Lydia

The author thanks the members of his supervisory committee, Dr. R. G. Selfridge, Dr. A. R. Bednarek, Dr. M. E. Thomas, and Dr. K. N. Sigmon, for their assistance and encouragement. Particular thanks are due Professor Selfridge, the committee chairman, and Professor Bednarek, the chairman of mathematics, for their many helpful suggestions.

This paper was typed by the ATS/360. This research was supported in part by a Graduate Fellowship from The University of Florida and in part by a Traineeship from the National Aeronautics and Space Administration.

TABLE OF CONTENTS

Key to Notation

$f: A\text{--}B$	f is a mapping from a set A into a set B
$(h,j):(S,X,.)\text{--}(T,W,.)$	$h:S\text{--}T$, $j:X\text{--}W$
$(h,j,k):(S,X,.,Y,*)\text{--}(T,W,.,Z,*)$	$h:S\text{--}T$, $j:X\text{--}W$, $k:Y\text{--}Z$
$A^{\underline{u}}B$	A is isomorphic to B
$(S,X,.)^{\underline{u}}(T,W,.)$	$(S,X,.)$ is semiautomaton isomorphic to $(T,W,.)$
$(S,X,.,p)^{\underline{u}}(T,W,.,q)$	$(S,X,.,p)$ is semimachine isomorphic to $(T,W,.,q)$
$(S,X,.,Y,*)^{\underline{u}}(T,W,.,Z,*)$	$(S,X,.,Y,*)$ is automaton isomorphic to $(T,W,.Z,*)$
$E(T)$	input right congruence relative to a subset T of a state space
$K(j,R)$	quasikernel of a mapping j and a relation R
$K(j)$	kernel of a mapping j
$\underline{e}$	the equivalence class containing the identity of a semigroup
$H1(X)$	semiring on the power set of a semigroup X with union for addition and elementwise multiplication
$Hn(X)$	semiring of matrices over $H1(X)$ with analogous matrix operations
$\underline{b}$	the subscript of 1 in b where b is an elementary basis element of a vector space of n-tuples

Abstract of Dissertation Presented to the Graduate Council
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy
at the University of Florida

THEOREMS FOR FINITE AUTOMATA

By

Reverdy Edmond Wright

June 1971

Chairman: Ralph G. Selfridge
Major Department: Mathematics
Minor Department: Industrial and Systems Engineering

The automata of this paper are finite in that they have a finite state space and finitely generated semigroups. Definitions of these systems and useful mappings between them are developed and used to examine their properties.

Automata without output are shown to be isomorphic to some with state spaces that are right congruences when a start state is designated or mappings from subsets of the integers into the integers for state spaces otherwise. A semiring of matrices over a semiring is used in the development of the latter.

Automata with output are shown to be homomorphic to some with state spaces that are sets of functions from the input semigroup to mappings from input to output. These homomorphisms involve identity mappings on the input and output and are shown to indicate "black box" equivanence between automata.

The idea of multiprogramming is defined without recourse to products of automata. Certain products are shown to be multiprogramming automata. In some cases, the product is the minimal automaton which can accomplish a given multiprogramming task.

The ability of a given automaton to act as a multiprogramming automaton is investigated. The existence of input elements satisfying the condition that, except for the identity, no power of one is a power of the other is shown to be necessary but not sufficient. A further condition is shown to be sufficient.

In the case of free semigroups, it is shown that the generators of the input semigroups need not be the same in the simulated automaton as in the multiprogramming automaton.

INTRODUCTION

This dissertation treats a special class of automata, finite sequential automata with finitely generated semigroups of input and output. Because the terms used in the Theory of Sequential Automata are defined differently by various authors, they must be defined at the outset of this paper. Although all semigroups are assumed herein to have identity elements, this fact will be emphasized by referring to them as monoids from time to time. No topology need be assumed for any semigroup or state space.

The topic is developed by starting with a class (semimachines) of sequential automata with designated start states and no output. The development continues with an investigation of the class (semiautomata) of sequential automata with neither start state nor output. In each case, homomorphisms are investigated and canonical forms developed.

A semiring of matrices with subsets of semigroups for entries is generated during the investigation of semiautomata. Each semiautomaton is shown to have a finite subset of these matrices corresponding to it. Any two semiautomata with the same input semigroup are isomorphic if and only if they correspond to the same set of these matrices.

In Chapter 3, the classes (automata and machines) of sequential automata with output, first without and then with designated start states are investigated. Homomorphism is shown to be a sufficient condition for "black-box" equivalence of sequential automata. Canonical forms are developed for minimal sequential automata, those with the smallest number of states which have given input to output properties. Finally, with the help of the canonical forms for semimachines and semiautomata, canonical forms are developed for those automata and machines which are not necessarily minimal.

In Chapter 4, the idea of multiprogramming for sequential automata is explored; first for automata and machines as developed in the previous chapters and then for the special case where the input semigroup is a finitely generated free monoid. The notion of product is not used in the definition, but product automata are shown to be examples of multiprogramming automata. The principal result might loosely be stated, "The most efficient way to build a machine to do the job of two machines is to put these two machines into a single box unless there is a more efficient way to build at least one of the two machines." Both necessary conditions and sufficient conditions for a machine to be a multiprogramming machine are investigated. An example showing that, even in the case of free semigroups of input and output, the generators of semigroups for simulating and simulated machines need not be the same is also important.

CHAPTER 1: PRELIMINARIES

Definition 1.01. A Semiautomaton is a triple (S,X,.) consisting of a non-empty finite set S, a finitely generated semigroup X, and a function . from SxX into S with the property that s.(xy)=(s.x).y for all s in S and all x and y in X. The set S is called the state space. The semigroup X is called the input semigroup. Frequently a finite set of generators of X will be selected and called the input alphabet. In particular the generators of a free monoid are thus designated. When an input alphabet is designated, each generator other than the identity is said to have length 1, the identity is said to have length 0, and length is defined for all other elements of X by the shortest product of generators. The function is called the transition function and its property, s(xy)=(sx)y, the sequential property. The image under the transition function will usually be denoted by juxtaposition.

Definition 1.02. A Semimachine is a quadruple (S,X,.,r) where (S,X,.) is a semiautomaton and r is an ingress, a state such that, for all s in S, there exists an x in X such that s=r.x. There may be other ingresses but only the one designated is the start state.

Definition 1.03. An Automaton is a quintuple (S,X,.,Y,*) where (S,X,.) is a semiautomaton, Y is a semigroup and the output function $*$ is a function from SxX into Y with the property that, for all s in S and all x and z in X, $s*(xz)=(s*x)(sx*z)$ and Y is the monoid generated by the range of the output function.

Definition 1.04. A Machine is a sextuple $(S,X,.,r,Y,*)$ where $(S,X,.,Y,*)$ is an automaton and $(S,X,.,r)$ is a semimachine.

The definitions of the systems are generalizations of those used by S. Ginsburg (G1) and A. Ginzburg (G2 and G3). Ginzburg's definition of semiautomaton is restricted to finitely generated free monoids. Ginsburg's complete sequential machine and Ginzburg's Mealy machine or Mealy automaton are essentially this paper's automaton restricted to finitely generated free semigroups of input and either free monoids or right zero semigroups of output. Ginsburg's quasimachine is still more general in that its state space may be infinite. His abstract machine is less general in that its output semigroup must be left-cancellative. (G2 and G3) (G1) (G2 and G3) (G1)

Ginzburg's cyclic semiautomaton and its generator are restrictions of this paper's semimachine and its start state. An extension analogous to that from semiautomaton to automaton leads to the definition of machine.

Definition 1.05. A Semiautomaton Homomorphism is a pair of mappings (h,j) such that $h:S--T$, $j:X--Z$, j is a semigroup homomorphism, and $(sx)h=(sh)(xj)$. 'f both h and j are one-to-one and onto, then the pair (h,j) is a semiautomaton isomorphism.

Definition 1.06. A Semimachine Homomorphism is a semiautomaton homomorphism (h,j) such that the image of the start state of the first machine is that of the second. If (h,j) is a semimachine homomorphism and a semiautomaton isomorphism, then it is a semimachine isomorphism.

Definition 1.07. An Automaton Homomorphism from (S,X,.,Y,*) to (T,Z,.,W,*) is a triple of mappings (h,j,k) such that (h,j) is a semiautomaton homomorphism and such that k:Y--W is a semigroup homomorphism and (s*x)k=(sh)*(xj) for all s in S and all x in X. Notice that neither the sequential property on the output nor that k is a semigroup homomorphism implies the other. If k is a semigroup isomorphism onto W and (h,j) is a semiautomaton isomorphism, then (h,j,k) is an automaton isomorphism.

Definition 1.08. A Machine Homomorphism is an automaton homomorphism (h,j,k) such that (h,j) is a semimachine homomorphism. If (h,j) is a semimachine isomorphism and (h,j,k) is an automaton isomorphism, then (h,j,k) is a machine isomorphism. When the meaning is clear from the context, i will frequently be used to denote the identity mapping on one of the state spaces or semigroups. For example, (i,j,i): (S,X,.,Y,*)--(S,Z,.,Y,*) would mean a triple consisting of the identity mapping on S, a mapping from X into Z, and the identity mapping on Y.

Homomorphism is defined for semiautomata by Ginzburg (G3) in an analogous fashion. It has also been defined by others (e. g. Norris (N)) for other types of automata with only inputs and states. Isomorphism, but not homomorphism, is defined for automata with output by Hartmanis and Stearns (HS). Ginsburg (G1) likewise defines only the former. Arbib (KFA) defines homomorphism slightly differently but makes little use of his definition.

Definition 1.09. A Subsemiautomaton of a semiautomaton (S,X,.), semimachine (S,X,.,r), automaton (S,X,.,Y,*), or machine (S,X,.,r,Y,*) is a semiautomaton (S',X',.) such that S' is a subset of S, X' is a finitely generated subsemigroup of X, and the transition functions agree on the elements of S'xX' and the restricted range is within S'.

Definition 1.10. A Subsemimachine of one of these same systems is a semimachine (S',X',.,q) where q is some element of S (not necessarily r), X' is a finitely generated subsemigroup of X, and the transition functions agree on S'xX'.

Definition 1.11. A Subautomaton of an automaton (S,X,.,Y,*) or machine (S,X,.,r,Y,*) is an automaton (S',X',.,Y',*) such that (S',X',.) is a subsemiautomaton, the output functions agree on S'xX', and the range of the restricted output function generates Y'.

Definition 1.12. A Submachine of an automaton or machine is a machine the automaton of which is a subautomaton and the semimachine of which is a subsemimachine.

Definition 1.13. The Internal Product Semiautomaton of the finite set of semiautomata (Si,Xi,.i) where i=1,...,n, and the Xi are subsemigroups of a semigroup X such that, for any i, the mapping j which maps each element of Xi onto itself but maps all other elements of the union of the Xi onto the identity can be extended to a semigroup homomorphism, is a semiautomaton (T,W,.) where T is the Cartesian product of the Si, W is the subsemigroup of X generated by the union of the Xi, and the transition function is defined for each element x of the union of the Xi by coordinatewise transition on all coordinates where x is in the respective Xi and by the identity transition on all other coordinates and the transition function is defined for all other elements by the sequential property. The internal product semiautomaton of a set of semimachines, automata, or machines is defined as that of their semiautomata.

If each input monoid Xi is disjoint from each of the others, this product is essentially Ginzburg's direct product (G3). If there is a pair of input monoids with elements other than the identity in their intersection, the effect of an input from this intersection is seen in more than one of the coordinates. Of particular interest is the case when X=X1=...=Xn when each input may affect every coordinate. An example of this is seen in the representative semimachine defined below.

Definition 1.14. The Internal Product Semimachine of the finite set of semimachines (Si,Xi,.i,qi), where i=1,...,n and the Xi have the same property as in the previous definition, is the semimachine (T,W,.,q) where (T,W,.) is the appropriate subautomaton of the internal product automaton of the semimachines and the start state q is (q1,...,qn).

Definition 1.15. The Internal Product Automaton of the finite set of automata (Si,Xi,.i,Yi,*i), where i=1,...,n, the Xi have the same property as in the previous definition, and the Yi are isomorphic to subsemigroups of a semigroup Y, is the automaton (T,W,.,Z,*) where (T,W,.) is the internal product semiautomaton of the semiautomata (Si,Xi,.i) and Z is the subsemigroup of Y generated by the image of TxX under the mapping * defined by extending the *i such that (...,si,...)*x corresponds to (si)(*i)x for all x in Xi, provided the output sequential property holds. The internal product automaton of a set of machines is that of their automata.

Definition 1.16. The Internal Product Machine of the finite set of machines (Si,Xi,.i,qi,Yi,*i), where i=1,...,n and the Xi and Yi have the same properties as in the previous definition, is the machine (T,W,.,q,Z,*) where (T,W,.,q) is the internal product semimachine and (T,W,.,Z,*) is a subautomaton of the internal product automaton.

Definition 1.17. The Representative Semimachine of a semiautomaton is the internal product semimachine of the subsemimachines of the semiautomaton starting with each of its states in turn.

Definition 1.18. The Representative Machine of an automaton is the internal product machine of the submachines of the automaton starting with each of its states in turn where the output semigroup is the Cartesian product of the output semigroups of the respective submachines.

CHAPTER 2: SEMIMACHINES AND SEMIAUTOMATA

Section 1: General Discussion

The presentation in this chapter does not follow the development usually made for semiautomata with finitely generated free semigroups of input but does touch on familiar results. The usual treatment of homomorphisms is either to restrict the semigroup homomorphism on the input to a semigroup isomorphism or to the homomorphism on free semigroups induced by a mapping of generators into generators.

The relations on the input semigroup and equivalence classes are frequently mentioned in connection with the class of regular subsets of a semigroup. The classic works for regular subsets of finitely generated free semigroups are those of Kleene (K) and of Rabin and Scott (RS). McKnight (Mc) has generalized Kleene's results. A very lucid treatment of regular sets is made by Ginzburg (G3).

Definition 2.01. The Input Right Congruence relative to a subset T of the state space S of a semimachine or semiautomaton is an equivalence relation E(T) on the input semigroup such that (x,z) is in E(T) when tx=tz for all t in T. Clearly this is a right congruence relation.

Proposition 2.02. If T is a subset of T', then E(T') is contained in E(T).

Proof: Follows immediately from the definition. The converse does not necessarily hold.

Definition 2.03. The Input Congruence of a system with state space S is E(S).

Definition 2.04. The Input Right Congruence of a semimachine with start state r is E(r), the input right congruence with respect to the singleton set of the start state r. Notice that any subset T of the state space S which contains an element s has the property that E(T) contains E(S) and is contained in E(s). E(S) is clearly the intersection over all s in S of the E(s).

Definition 2.05. For any mapping j from a set X to a set Y and any relation R on Y, the set of all pairs (x,x') in X with the property that (xj,x'j) is in R is called K(j,R), the Quasikernel of j and R, or K(j), the Kernel of j if R is the identity relation. Morris (N) defines the kernel in this way. The definition of quasikernel is an obvious extension.

Proposition 2.06. For any semimachine (S,X,.,r) or semiautomaton (S,X,.) the pair of mappings (i,j), where i is the identity mapping on S, j is the natural mapping from the elements of X to the elements of X/E(S), and the transition function is the naturally induced s(xj)=sx, is a semimachine or semiautomaton homomorphism.

$$\begin{tikzcd}
SxX \arrow[r, "."] \arrow[d, "{(i,j)}"'] & S \arrow[d, "i"] \\
Sx(X/E(S)) \arrow[r, "."] & S
\end{tikzcd}$$

Proof: If (y,y') is in E(S), then for all s in S sx is also in S and (sx)y=(sx)y'. Hence E(S) is a congruence relation, X/E(S) is a semigroup, and j is a semigroup homomorphism.

This result is a special case of a result of Bednarek and Wallace (BW2). The semigroup of transformations induced on S by X is isomorphic to X/E(S) and is called by Ginzburg (G2) the Semigroup of the semimachine or semiautomaton.

<u>Proposition</u> <u>2.07</u>. If (h,j) and (h',j') are semimachine or semiautomaton homomorphisms from (S,X,.,r) to (S',X',.,r') and thence to (S",X",.,r") respectively or from (S,X,.) to (S',X',.) and thence to (S",X",.), then (hh',jj') is a semimachine or semiautomaton homorphism from (S,X,.,r) to (S",X",.,r) or from (S,X,.) to (S",X",.).

$$\begin{tikzcd}
SxX \arrow[rrr, "."] \arrow[dr, "{(h,j)}"] \arrow[dd, "{(h'',j'')}"'] & & & S \arrow[dl, "h"'] \arrow[dd, "h''"] \\
& S'xX' \arrow[r, "."] \arrow[dl, "{(h',j')}"] & S' \arrow[dr, "h'"'] & \\
S''xX'' \arrow[rrr, "."] & & & S''
\end{tikzcd}$$

Proof: (sx)(hh')=((sx)h)h'=((sh)(xj))h'=((sh)h'((xj)j') =(s(hh'))(x(jj')) and in the case of the semimachine homomorphism r(hh')=(rh)h'=r'h'=r". Norris (N) proved this for acts as a corollary to a lemma.

<u>Proposition</u> <u>2.08</u>. If (h,j):(S,X,.,q)--(T,Y,.,r) is a semimachine homomorphism, then (h,i) is a semimachine homomorphism from (S,X,.,q) to (T,X,.,r) if the transition function on the latter is defined by (sh)x=(sx)h and further (i,j):(T,X,.,r)--(T,Y,.,r) is a semimachine homomorphism. Likewise, there is a decomposition of a semiautomaton homomorphism.

$$\begin{tikzcd}
SxX \arrow[rrr, "\cdot"] \arrow[dr, "{(h,i)}"] \arrow[dd, "{(h,j)}"'] & & & S \arrow[dl, "h"'] \arrow[dd, "h"] \\
& TxX \arrow[r, "\cdot"] \arrow[dl, "{(i,j)}"] & T \arrow[dr, "i"] & \\
TxY \arrow[rrr, "\cdot"] & & & T
\end{tikzcd}$$

Proof: ((sh)x)i=(sh)x=(sx)h=(sh)(xj)=((sh)i)(xj) and in the case of the semimachines qh=r and ri=r.

<u>Proposition</u> <u>2.09</u>. If (i,j):(S,W,.)--(S,X,.) is a semiautomaton homomorphism and j maps W onto X and i is the identity mapping on S, then W/E(S)≅X/E(S) and (S,W/E(S),.)≅(S,X/E(S),.).

$$\begin{tikzcd}
SxW \arrow[rr, "\cdot"] \arrow[dr, "{(i,j)}"] \arrow[dd, "{(i,j'')}"'] & & S \arrow[dr, "i"] \arrow[dd, "i"'] & \\
& SxX \arrow[rr, "\cdot"] \arrow[dd] & & S \arrow[dd, "i"] \\
Sx(W/E(S)) \arrow[rr, "\cdot"] \arrow[dr, "\cong"', "{(i,j')}"] & & S \arrow[dr, "i"] & \\
& Sx(X/E(S)) \arrow[rr, "\cdot"] & & S
\end{tikzcd}$$

Proof: Let j'' be the natural mapping from W to $W/E(S)$ and j' the natural mapping from X to $X/E(S)$. Further let w and w' be in W. If (w,w') is in $E(S)$ then $s(w(jj'))=s((wj)j')=s(wj)=sw=s(wj'')=s(w'j'')=\ldots=s(w'(jj'))$ for all s in S. If (w,w') is not in $E(S)$, then for some s in S $s(w(jj'))=s(wj'')\neq s(w'j'')=s(w'(jj'))$.

Since j, j', and j'' are all onto there is exactly one element of $X/E(S)$ corresponding to any v in $W/E(S)$ and that is $w(jj')$ where w is any element of W such that $wj''=v$. Now $s(w(jj'))=s(wj'')=sv$.

Section 2: Semimachines

A semimachine has the simplest structure of all of the systems here described. It is a basic part of all systems since for each state q of any system there is a corresponding subsemimachine (S',X,.,q).

Definition 2.10. The Right Congruence Semimachine is defined for any right congruence relation E with a finite set of equivalence classes on a semigroup X to be a semimachine (T,X,.,$\underline{e}$) such that the states are equivalence classes of E, the start state $\underline{e}$ is the equivalence class containing the identity element e of X, and the transition function takes ($\underline{e}$,x) to the equivalence class containing x and (s,x) to the equivalence class containing zx where z is any element of s.

Proposition 2.11. For any semimachine (S,X,.,r), the pair of mappings (h,i), where i is the identity on X and h maps each s in S to the set of all x such that rx=s, is a semimachine isomorphism from (S,X,.,r) to the right congruence semimachine of E(r).

$$\begin{tikzcd}
S\mathrm{x}X \arrow[r, "\cdot"'] \arrow[d, "\underline{\underline{u}}\ (h{,}i)"'] & S \arrow[d, "h"'] \\
(X/E(r))\mathrm{x}X \arrow[r, "\cdot"'] & Y/E(r)
\end{tikzcd}$$

Proof: For all x in X, x is an element of $(rx)h$. If x is an element of both sh and th, then $s=rx=t$. x and y are elements of sh if and only if $rx=s=ry$. Since $re=r$, e is an element of rh. If z is an element of sh, then since z is an element of sh zx is an element of $(sh)x$ and since $sx=(rz)x=r(zx)$ zx is an element of $(sx)h$.

Proposition 2.12. Let E and E' be right congruence relations with finite numbers of equivalence classes over a finitely generated semigroup X. Let i be the identity mapping on X. If E is a subset of E', then there is a semimachine homomorphism (h,i) from the right congruence semimachine of E' to that of E. If E is not a subset of E', then no pair of mappings (h,i) is a semimachine homomorphism from the right congruence semimachine of E to that of E'.

Proof: Assume (x,y) is in E but not in E'. The equalities $(rx)h=(rh)x$ and $(ry)h=(rh)y$ must hold and rh must be the start state of its semimachine for (h,i) to be a semimachine homomorphism, but $(rh)x\neq(rh)y$ while $rx=ry$. Assume E is a subset of E'. Since X/E refines X/E', h can be defined such that s is a subset of sh for any s in X/E. Clearly e is an element of both $\underline{e}$ and $\underline{e}h$. Since the transition function of each machine is defined by right multiplication, sx is a subset of $(sh)x$. Since sx is a subset of $(sx)h$ and X/E' is a partition, $(sh)x=(sx)h$.

Proposition 2.13. For any semimachine $(S,X,.,r)$ there exists a mapping h from $X/E(S)$ to S such that the pair (h,i) is a semimachine homomorphism from the right congruence semimachine of the input congruence where i is the identity on X. Further, for any x in an equivalence class t, rx=th.

$$\begin{tikzcd}
(X/E(S))xX \arrow[r, "\cdot"'] \arrow[d, "{(h,i)}"'] & X/E(S) \arrow[d, "h"] \\
SxX \arrow[r, "\cdot"'] & S
\end{tikzcd}$$

Proof: Since E(S) is contained in E(r), by proposition 2.12 there is a semimachine homomorphism (h',i) from (X/E(S),X,.,$\underline{e}$) to (X/E(r),X,.,$\underline{e}$). From proposition 2.11 there is an isomorphism (h",i) from (X/E(r),X,.,$\underline{e}$) to (S,X,.,r). By proposition 2.07 (h,i)=(h'h",ii) is the required semimachine homomorphism. Since any t in X/E(S) is a subset of th' in X/E(r), any x in t is also in th' which is the set of all z such that rz=th'h". Hence rx=th.

The representative machine of (S,X,.,r) is clearly isomorphic to the machine (X/E(S),X,.,$\underline{e}$). The representative machine is maximal in the sense that any machine having the same semigroup of transformations is one of its homomorphic images. Although there is at least one machine with as few states as any other with the same semigroup of transformations, there may be none for which some semimachine homomorphism can be found from any other with the same semigroup.

Proposition 2.14. Let (S,X,.,q) and (T,X,.,r) be two semimachines with the same input semigroup. The input right congruence of $(S,X,.,q)$ refines that of $(T,X,.,r)$ if and only if there exists a semimachine homomorphism (h,i) from $(S,X,.,q)$ to $(T,X,.,r)$ with i the identity on X.

Proof: By proposition 2.11, each machine is isomorphic to the right congruence semimachine of its input right congruence. Since semimachine homomorphisms compose, there is a semimachine homomorphism from one semimachine to the other if and only if there is a corresponding semimachine homomorphism between right congruence semimachines of their input right congruences. This, in turn, is equivalent to the refinement of $E(r)$ by $E(q)$.

<u>Proposition</u> <u>2.15</u>. Let $E(q)$, $E(r)$, and $E'(r)$ be the respective input right congruences of the machines $(S,X,.,q)$, $(T,Y,.,r)$, and $(T,X,.,r)$ of proposition 2.08. $E'(r)$ is $K(j,E(r))$.

Proof: For any x and z in X, $rx=(qh)x=(qx)h=(oh)(xj)=r(xj)$. and $rz=(qh)z=(qz)h=(qh)(zj)=r(zj)$. Hence $rx=rz$ if and only if $r(xj)=r(zj)$.

<u>Proposition</u> <u>2.16</u>. Let $(S,X,.,q)$ and $(T,Y,.,r)$ be semimachines. If there is a semimachine homomorphism (h,j) from the former onto the latter, then the input right congruence of the former refines the quasikernel of j and the input right congruence of the latter. If there is a semigroup homomorphism from X onto Y such that the input right congruence of the former refines the quasikernel of j and the input right congruence of the latter, then there

is a mapping h from S onto T such that (h,j) is a semimachine homomorphism.

Proof: Assume that there is a semigroup homomorphism j such that E(q) refines the quasikernel of j and E(r). The mapping h defined by (rx)h=q(xj) for each rx in S is the mapping such that (h,j) is a semimachine homomorphism. The remainder of the proof follows directly from propositions 2.08, 2.14, and 2.15.

Proposition 2.17. If (i,j):(S,W,.,r)--(S,X,.,r) is a semimachine homomorphism, then W/E(S)$\underline{u}$X/E(S) and (W/E(r),W/E(S),.,$\underline{e}$)$\underline{u}$(X/E(r),X/E(S),.,$\underline{e}$).

$$\begin{tikzcd}
SxW \arrow[rr, "\cdot"] \arrow[dr, "{(i,j)}"] \arrow[dd] & & S \arrow[dr, "i"] \arrow[dd] & \\
& SxX \arrow[rr, "\cdot"] \arrow[dd] & & S \arrow[dd] \\
(W/E(r))x(W/E(S)) \arrow[rr, "\cdot"] \arrow[dr, "\underline{u}"] & & W/E(r) \arrow[dr, "\underline{u}"] & \\
& (X/E(r))x(X/E(S)) \arrow[rr, "\cdot"] & & X/E(r)
\end{tikzcd}$$

Proof: Since any state in S can be expressed as rw or rx j is necessarily onto. The rest of the proof follows directly from 2.09 and 2.11.

Right congruence semimachines can be considered canonical forms for semimachines. Each semimachine is isomorphic to the right congruence semimachine of its input right congruence. All of the properties of a semimachine qua semimachine are those which are preserved by the isomorphism from a semimachine to its canonical form.

Proposition 2.16 gives a criterion for the existence of homomorphisms in terms of right congruences.

Section 3: Semiautomata

A semiautomaton with at least one ingress can be investigated by treating it as a machine with one of its ingresses as start state. Decomposing a semiautomaton into semimachines does not preserve all of the characteristics of a semiautomaton unless correspondences are defined between the states of each semimachine and the set of semimachines.

Consider the following illustration of the difficulty: Let X be the commutative semigroup with three elements such that $ee=e$, $ea=ab=a$, and $eb=aa=bb=b$. Let the transition function for the machines $(\{1,2,3,4\},X,.)$ and $(\{5,6,7,8\},X,.)$ be given by the Cayley tables:

	e	a	b
1	1	2	1
2	2	1	2
3	3	1	2
4	4	1	2

	e	a	b
5	5	6	5
6	6	5	6
7	7	6	5
8	8	5	6

$\{\{e,b\},\{a\}\}=X/E(1)=X/E(2)=X/E(5)=X/E(6)$,

$\{\{e\},\{a\},\{b\}\}=X/E(3)=X/E(4)=X/E(7)=X/E(8)$.

But no isomorphism exists since $2a=3a=4a\neq 1a$ while $6a=8a\neq 7a=5a$.

In pursuit of a canonical form similar to the right congruence semimachine it will be convenient to examine the following system expressed in terms of finite dimensional vector spaces:

Let B be a basis of a finite dimensional vector space V and A be a semigroup of linear transformations of V with the property that, for all (b,a) in BxA, ba is in B. Clearly $(B,A,.)$ is a semiautomaton.

Proposition 2.18. For any semiautomaton $(S,X,.)$ with n states, any basis B of an n-dimensional vector space V, and any one-to-one mapping h from S to B, the mapping j from X into the set of linear transformations on V defined by $(sh)(xj)=(sx)h$ is a semigroup homomorphism. A fortiori: (h,j) is a semiautomaton homomorphism and the semiautomaton homomorphism $(h,i):(S,X,.)\text{--}(B,X,.)$ is a semiautomaton isomorphism.

Proof: For each x in X there is a unique linear transformation carrying each sh of B to $(sx)h$. Let j be the mapping which carries each x in X to this transformation. That is to say: $(sx)h=(sh)(xj)$. For any x and z in X, $(xj)(zj)=(xz)j$ since for each s in S $(sh)(xj)(zj)=((sx)h)(zj)=((sx)z)h=(s(xz))h=(sh)((xz)j)$.

The existence of the semiautomaton homomorphism (h,i) follows from proposition 2.08. Since h is one-to-one and onto, (h,i) is a semiautomaton isomorphism.

When the vector space is the set of n-tuples over a field and the basis is the elementary basis, then the matrices of the linear transformations give a peculiarly graphic picture of the semiautomaton. This representation is well known in some circles. (A) In this case, multiplication of an n-tuple by a matrix amounts to selecting a row of the matrix since each n-tuple consists of a single 1 and several 0's. Hence, each row of every matrix must be a member of the elementary basis. Since the only elements of the field that are used in this construction are the zero and the one, the set of n-tuples over any semiring with zero and one which corresponds to the elementary basis of a vector space and these same linear transformations are isomorphic to this construction.

In this case the mapping j is a representation of X by a Rees matrix semigroup over a group with zero since the set with elements 0 and 1 under multiplication is such and only one element of each row is nonzero. The representation is faithful when $X \cong X/E(S)$.

The following semiring of matrices over a semiring of subsets of a semigroup provides a useful tool for the next step in the development of a canonical semiautomaton. Consider the power set of a semigroup X under the operations of union and setwise multiplication H1(X). Both operations are associative. Union is commutative. Multiplication distributes over union. The empty set is the zero and the singleton set of the identity of X is the identity of H1(X).

Definition 2.19. For any monoid X, H1(X) is thus defined.

Since the distributivity of the usual matrix product over addition depends only on the above properties of a semiring, the n by n matrices Hn(X) over H1(X) with union defined coordinatewise and multiplication defined by $AB(i;k)=\bigcup A(i;j)B(j;k)$ is a semiring.

Definition 2.20. Hn(X) is thus defined.

Kameda (Ka) has independently developed several matrices over sets which relate regular expressions of input to regular events over the output alphabet. One of these matrices resembles matrices of Hn(X).

Investigation of all of the properties of Hn(X) is beyond the scope of this paper. Some properties which are not necessary for proofs will be mentioned to provide a more graphic picture of the development of a canonical form. Definitions may be considerably more general than absolutely necessary.

Definition 2.21. Multiplication of an n-tuple s over H1(X) and a matrix z in Hn(X) is defined such that sz(j) is the union of the s(i)z(i;j) where i=1,...,n. This is analogous to multiplication of a vector by a matrix when both are over a field.

Proposition 2.22. A matrix P is a right unit if and only if every entry except one in each row and in each column is the empty set and every nonempty entry is a set of right units which have a common right inverse.

Proof: Let Q be a right inverse of P. Each diagonal entry of PQ is the singleton set of the identity of X. Hence for each i there must be a j such that P(i;j)Q(j;i) is this singleton set; hence neither P(i;j) nor Q(j;i) can be empty, but rather all of the elements of P(i;j) must be left inverses of all the elements in Q(j;i). Since for all k≠i P(k;j)Q(j;i) must be empty and Q(j;i) is not empty P(k;j) must be empty. Since the number of rows is the same finite number as the number of columns, only one j can correspond to a single i in the previous statements.

In H1(X) there will be one-sided units if X has one-sided units. In this case the definition of similarity of matrices usually formulated does not necessarily give an equivalence relation. For example, if X is the bicyclic semigroup with ab=e≠ba and y is said to be similar to x if there are elements p and q such that pq=e and xq=qy. Since p must be some power of a and q must be the same power of b, the set of elements of X similar to any given x will include x, axb, aaxbb, etc. For any x in X, x is similar to bxa but bxa is not similar to x. To avoid this difficulty the definition of similarity will be restricted to two-sided units.

<u>Definition</u> <u>2.23</u>. The matrices A and B in Hn(X) are Similar if there are matrices P and Q in Hn(X) such that PQ=QP=I where I is the identity of Hn(X) and AQ=QB.

It should be noted that Hn(X) has no one-sided units if X has none. This is because

$P(i;j)Q(j;i)=Q(j;i)P(i;j)$ is either empty or the singleton set of the identity of X.

Definition 2.24. The matrices A and B in $Hn(X)$ are Strictly Similar if there are matrices P and Q in $Hn(X)$ such that each nonempty entry of P is the singleton set of the identity of X and $PQ=I$ and $A=PBQ$. Clearly $B=QAP$ and Q is the transpose of P. Strictly similar implies similar. If, as in the case of free semigroups, there are no units other than the identity, then strict similarity and similarity are equivalent.

Definition 2.25. In $Hn(X)$, the partial order $\leq$ is defined by $A\leq B$ when $A\cup B=B$.

Proposition 2.26. If $A\leq B$, then $AC\leq BC$ and $CA\leq CB$.

Proof: $(AC)\cup(BC)=(A\cup B)C=BC$ and $(CA)\cup(CB)=C(A\cup B)=CB$.

Proposition 2.27. Let $(S,X,.)$ be a semiautomaton. Let B be the elementary basis of n-tuples over a field. Let j be a semigroup homomorphism as defined in proposition 2.18. Define $m':X\text{--}Hn(X)$ such that every entry of xm' is an empty set or a singleton set of x as the corresponding entry of xj is a zero or one respectively. m' is a semigroup homomorphism from X into the multiplicative semigroup of $Hn(X)$.

Proof: $(xy)m'$ has singleton sets of xy and empty sets in exactly those places that $(xy)j$ has ones and zeros respectively. The product of xm' and ym' has singleton sets of xy and empty sets in exactly those places that the product of xj and yj has ones and zeros respectively. Since (xy)j=(xj)(yj), it must follow that (xy)m'=(xm')(ym').

<u>Proposition</u> <u>2.28</u>. Let (S,X,.) be a semiautomaton. Let m' be the semigroup homomorphism defined in proposition 2.27. Let m:H1(X)--Hn(X) be defined so that for any non-empty subset a of X $am=U\{xm' | x\varepsilon\ a\}$and m maps the empty set to the n-by-n matrix of empty sets. m is a semiring homomorphism.

Proof: $(aUb)m = U\{xm' | x\varepsilon(aUb)\} = (U\{xm' | x\varepsilon a\})U(U\{xm' | x\varepsilon b\}) =(am)U(bm)$. Since (xy)m'=(xm')(ym') by proposition 2.27, $(ab)m= \{xm' | x\varepsilon ab\} = \{(xy)m' | x\varepsilon a, y\varepsilon b\} = \{(xm')(ym') | x\varepsilon a, y\varepsilon b\} =\{xm' | x\varepsilon a\}\{ym' | y\varepsilon b\}=(am)(bm)$.

<u>Proposition</u> <u>2.29</u>. Let (S,X,.) be a semiautomaton. Let G be a finite set of generators for X. Let m and m' be defined as in proposition 2.28. Define M(0) to be the image under m of the identity element of X. Define M(n) to be the image under m of the set of all words in X of length less than or equal to n with respect to the input alphabet G. The n-th power of M(1) is M(n).

Proof: The case when n=1 is trivial. Assume the case for n-1 and calculate the product M(n-1)M(1). If x in X is of length n or less, then there is a y of length n-1 or less and a z of length 1 or less such that x=yz. $xm'\leq M(n)$ only if $ym'\leq M(n-1)$ and $zm'\leq M(1)$.

Conversely, if $ym'\leq M(n-1)$ and $zm'\leq M(1)$, then $(yz)m'\leq M(n)$. M(n)=M(n-1)M(1).

Proposition 2.30. In the notation of propositions 2.28 and 2.29, Xm is the limit as n increases without bound of M(n).

Proof: Every x in X is a product of a finite sequence of generators. Therefore $Xm \leq UM(n)$. Since each M(n) is clearly contained in Xm, Xm=UM(n).

Proposition 2.31. Continuing the notation of propositions 2.27 through 2.30 and adding the notation: M=Xm and $\underline{b}$ is the integer such that $b(\underline{b})=1$, $M(\underline{sh};\underline{th})$ is the set of all x in X such that sx=t.

Proof: Multiplication by sh is equivalent to selecting the $\underline{sh}$-th row of a matrix. The $\underline{sh}$-th row of xj will be th if and only if sx=t. In other words the $\underline{th}$-th entry of the $\underline{sh}$-th row of xj is 1 and the same entry of the same row xm' is the singleton set of x if and only if sx=t. $M(\underline{sh};\underline{th})$ is by definition the union of the corresponding entries for all x in X but only those x for which sx=t have a non-empty entry in this position.

Proposition 2.32. Let k' now be defined to be any one-to-one mapping of the state space of (S,X,.) onto the integers 1 through n where n=|S|. Let k now be defined so that it is that mapping from S into the set of functions from X to the integers 1 through n such that the image of x under the mapping sk (notation: x:(sk)) is (sx)k'. The pair of mappings (k,i) where i is the identity mapping on X is a semiautomaton isomorphism from (S,X,.) to (Sk,X,.) with the naturally induced transition function (sk)x=(sx)k.

Proof: If $s \neq t$, then $(se)k' = sk' \neq tk' = (te)k'$ and hence $e:(sk) \neq e:(tk)$. The remaining properties of a semiautomaton isomorphism follow directly from the hypotheses.

Proposition 2.33. Let the notation be as in proposition 2.32. For all x and y in X, $(xy):(sk) = y:((sx)k)$.

Proof: $(xy):(sk) = (s(xy))k' = ((sx)y)k' = y:((sx)k)$.

Proposition 2.34. If k and k' are defined as in proposition 2.32 and $sk' = \underline{sh}$ in the notation of proposition 2.31, then x is an element of $M(\underline{sh};\ x:(sk)\ )$.

Proof: By proposition 2.32, $x:(sk) = (sx)k' = \underline{(sx)h}$. By proposition 2.31, x is an element of $M(\underline{sh};\ \underline{(sx)h})$.

Proposition 2.35. Let the mappings h and m and the matrix M be those defined in propositions 2.27, 2.28, and 2.29 relative to $(S,X,.)$ and h'', m'', and M'' be those relative to $(T,X,.)$. Let $n = |S| \geq n'' = |T|$. The existence of an n'' by n matrix P over $H1(X)$ and its transpose Q such that every column of P has $n''-1$ empty sets and one singleton set of the identity e of X for entries and $PMQ = M''$ is a necessary and sufficient condition for the existence of a mapping g from S onto T such that (g,i) is a semiautomaton homomorphism.

Proof: Assume that (g,i) is a semimachine homomorphism from $(S,X,.)$ onto $(T,X,.)$. Since g is onto and h and h'' are both onto and one-to-one, there is a mapping p from the first n positive integers to the first n'' such that $\underline{shp} = \underline{sgh''}$ for all s in S. Let P be defined so that $P(j,j'')$

is the singleton set of e if $j''p=j$ and is the empty set otherwise. Let x be any element of $M''(\underline{th''};\underline{vh''})$. There exists an element s of S such that $sg=t$. Since g is a semimachine homomorphism, $(sx)g=v$. The mapping p was defined so that $\underline{shp}=\underline{th''}$ and $\underline{sxhp}=\underline{vh''}$. x is an element of $M(\underline{sh};\underline{sxh})= P(\underline{th''};\underline{sh})M(\underline{sh};\underline{sxh})Q(\underline{sxh};\underline{vh''})\leq PMQ(\underline{th''};\underline{vh''})$. Hence $M''\leq PMQ$.

Assume now that x is not an element of $M(\underline{th''};\underline{vh''})$. If $sg=t$, then $(sx)g\neq v$ and $\underline{sxhp}\neq\underline{vh''}$. For all s such that $sg=t$, x is only in $M(\underline{sh};\underline{sxh})$ and $P(\underline{th''};\underline{sh})M(\underline{sh};\underline{sxh})Q(\underline{sxh};\underline{vh''})$ is empty.

Hence $PMQ\leq M''$. It has now been shown that $M''=PMQ$ if (g,i) is a semimachine homomorphism.

Next assume the existence of the matrix P with the stated properties. Let p be defined such that. $P(jp;j)$ is a singleton set for each $j\leq n$. Let g be defined such that $\underline{shp}=\underline{sgh''}$ for all s in S.

Since $P(\underline{shp};\underline{sh})M(\underline{sh};\underline{sxh}\ )Q(\underline{sxh};\underline{sxhp}) \leq M''(\underline{shp};\underline{sxhp})=M''(\underline{sgh''};\underline{sxgh''})$, $exe=x$ is an element of $M''(\underline{sgh''};\underline{sxgh''})$. Since x is in $M''(\underline{th''};\underline{vh''})$ only if $tx=v$, $(sg)x=(sx)g$ and hence (g,i) is a semiautomaton homomorphism.

<u>Proposition</u> <u>2.36</u>. Let the notation be as in proposition 2.35. M is strictly similar to M'' if and only if there is a mapping g from S to T such that (g,i) is a semimachine isomorphism.

Proof: This is the special case of proposition 2.35 where n=n". The mapping p is a permutation. The matrix P is a unit of Hn(X).

Any semiautomaton which is the image under the semiautomaton isomorphism (k,i) of proposition 2.32 can be considered a canonical form. The canonical form is unique up to permutations of 1 through n. The properties of a semiautomaton qua semiautomaton are preserved by the isomorphism from a semiautomaton to its canonical form. Even as the transition function of a canonical form of a semimachine could be determined from the states, the transition function of a canonical form of a semiautomaton can be determined from its states.

Proposition 2.34 ties each canonical form to a matrix over H1(X). Proposition 2.35 then provides a characterization of homomorphism in terms of these matrices. For semiautomata with different input semigroups, there must be a semigroup homomorphism j such that each equivalence class of the quasikernel of j and of the matrix of the image automaton is the corresponding entry of the matrix PMQ of the preimage automaton.

CHAPTER 3: MACHINES AND AUTOMATA

Section 1: General Discussion

The first difference between the results of this chapter and analogous results is that the definitions are more general. When the case of finitely generated free semigroups of input and output is treated, the egdulk automaton is isomorphic to a minimal automaton and also has in its definition all information about the transition and output functions. Ginsburg (G1) and Booth (B) make clear presentations of the more usual treatment of minimization. Two states s and t are equivalent if there are mappings c and c' with the property of c in definition 3.06 from their respective state spaces into the state space of some automaton such that $sc=sc'$.

Proposition 3.01. If (h,i,i) is a machine homomorphism from $(S,X,.,q,Y,*)$ onto $(T,X,.,r,Y,*)$ or an automaton homomorphism from $(S,X,.,Y,*)$ onto $(T,X,.,Y,*)$, then for each state s in S and each x in X, $s*x=(sh)*x$.

Proof: $(sh)*x=(sh)*(xi)=(s*x)i=s*x$.

Proposition 3.02. If (h,j,k) and (h',j',k') are machine or automaton homomorphisms from $(S,X,.,r,Y,*)$ to $(S',X',.,r',Y',*)$ and thence to $(S'',X'',.,r'',Y'',*)$ respectively or from $(S,X,.,Y,*)$ to $(S',X',.,Y',*)$ and thence to (S",X",.,Y",*), then (hh',jj',kk') is a machine or automaton homomorphism from (S,X,.,r,Y,*) to (S",X",.,r",Y",*) or from (S,X,.,Y,*) to (S",X",.,Y",*).

$$\begin{tikzcd}
S \arrow[dr, "h"] \arrow[dd, "h''"'] & & S \times X \arrow[ll, "\cdot"'] \arrow[rrr, "*"] \arrow[dr, "{(h,j)}"] \arrow[dd, "{(h'',j'')}"'] & & & Y \arrow[dl, "k"'] \arrow[dd, "k''"] \\
& S' \arrow[dl, "h"] & & S' \times X' \arrow[ll, "\cdot"'] \arrow[r, "*"] \arrow[dl, "{(h',j')}"] & Y' \arrow[dr, "k'"] & \\
S'' & & S'' \times X'' \arrow[ll, "\cdot"'] \arrow[rrr, "*"] & & & Y''
\end{tikzcd}$$

Proof: By proposition 2.07, (sx)(hh')=(s(hh'))(x(jj')) and in the case of the machine homomorphism r(hh')=r".

By the definitions, (s*x)(kk')=((s*x)k)k'=((sh)*(xj))k' =((sh)h')*((xj)j')=(s(hh'))*(x(jj')).

<u>Proposition</u> <u>3.03</u>. If (h,j,k):(S,X,.,q,Y,*)--(T,W,.,r,Z,*) is a machine homomorphism which is onto, then (i,i,k) is a machine homomorphism from (S,X,.,q,Y,*) onto (S,X,.,q,Z,#) if the output function on the latter is defined by s#x=(s*x)k and further (h,j,i):(S,X,.,q,Z,#)--(T,W,.,r,Z,*) is a machine homomorphism. There is a like decomposition of an automaton homomorphism.

$$\begin{tikzcd}
S \arrow[dr, "i"] \arrow[dd, "h"'] & & S \times X \arrow[ll, "\cdot"'] \arrow[rrr, "*"] \arrow[dr, "{(i,i)}"] \arrow[dd, "{(h,j)}"'] & & & Y \arrow[dl, "k"'] \arrow[dd, "k"] \\
& S \arrow[dl, "h"] & & S \times X \arrow[ll, "\cdot"'] \arrow[r, "*"] \arrow[dl, "{(h,j)}"] & Z \arrow[dr, "i"] & \\
T & & T \times W \arrow[ll, "\cdot"'] \arrow[rrr, "*"] & & & Z
\end{tikzcd}$$

Proof: (s#x)i=s#x=(s*x)k=(sh)*(xj) and in the case of machines qi=q and qh=r.

Proposition 3.04. If (h,j,i):(S,X,.,q,Z,#)--(T,W,.,r,Z,*) is a machine homomorphism which is onto, then (h,i,i) is a machine homomorphism from (S,X,.,q,Z,#) onto (T,X,.,r,Z,*) if the transition and output functions on the latter are defined by (sh)x=(sx)h and (sh)*x=s#x and further (i,j,i) is a machine homomorphism from (T,X,.,r,Z,*) onto (T,W,.,r,Z,*).

$$\begin{tikzcd}
S \arrow[rr, "."] \arrow[ddd, "h"'] \arrow[dr] & & S\times X \arrow[rrr, "*"] \arrow[ddd] \arrow[dr] & & & Y \arrow[dl] \arrow[ddd] \\
& S \arrow[d, "h"] \arrow[ddl] & & S\times X \arrow[ll, "."'] \arrow[r, "*"] \arrow[d, "{(h,i)}"] \arrow[ddl, "{(h,j)}"'] & Z \arrow[d, "i"] \arrow[ddr, "i"] & \\
& T \arrow[dl, "i"] & & T\times X \arrow[ll, "."'] \arrow[r, "*"] \arrow[dl, "{(i,j)}"] & Z \arrow[dr, "i"] & \\
T & & T\times W \arrow[ll, "."] \arrow[rrr, "*"] & & & Z
\end{tikzcd}$$

Proof: Proposition 2.08 provides the proof for the transition functions and, in the case of the machines, the start states. For the output functions (s#x)i=s#x=(sh)*x=(sh)*(xi) and ((sh)*x)i=(s#x)i=(sh)*(xj)=((sh)i)*(xj).

Definition 3.05. A machine (S,X,.,q,Y,*) is Minimal if there is no machine (T,X,.,r,Y,*) such that |T|<|S| and (qx)*y=(rx)*y for all x and y in X. A minimal machine is as small as any machine which, having read in x, maps y to (qx)*y.

Definition 3.06. An automaton (S,X,.,Y,*) is Minimal if there is no automaton (T,X,.,Y,*) with mapping c from S into T such that $|T|<|S|$ and $(sx)*y=((sc)x)*y$ for all s in S and all x and y in X.

Section 2: Automata

Let T be a finite set of mappings from a finitely generated semigroup X onto a finite set of functions from X to a finitely generated semigroup Y such that T has the following properties: Y is the semigroup generated by the union of the ranges of the functions in the range of the mappings in T. For each t in T and each x in X there is a t' in T such that, for all y in X, $yt'=(xy)t$. The image $(yz):xt$ of yz in X under the mapping xt is equal to the product in Y of $y:xt$ and $z:(xy)t$.

Definition 3.07. An Egdulk Automaton is an automaton $(T,X,.,Y,*)$ where T is a set of mappings as described above and the transition function . is defined such that, for all y in X $y(t.x)=(xy)t$ and the output function is defined by $t*x=x:et$.

The sequential property is demonstrated by: For all z in X, $z(t.(xy))=((xy)z)t=(x(yz))t=(yz)(t.x)=z((t.x).y)$.

The output sequential property is demonstrated by: Since $(ex)t = (xe)t = e(tx)$, $t*(xz)=(xz):et=(x:et)(z:(ex)t)=(t*x)(z:e(tx))=(t*x)(tx*z)$.

Proposition 3.08. Let $(S,X,.,Y,*)$ be an automaton. Let h be a mapping from S into the set of mappings from X into the set of functions from X into Y with the property that any x in X is mapped by sh to the function which maps any y in X to $y:x(sh)=(sx)*y$. (h,i,i) is an automaton homomorphism from $(S,X,.,Y,*)$ onto the egdulk automaton $(Sh,X,.,Y,*)$.

Proof: Each i is a semigroup homomorphism. For each s in S and x in X, $(s*x)i=(se)*x=x:e(sh)=(sh)*x$ and for each y and z in X $z:y((sx)h) = ((sx)y)*z = (s(xy))*z = z:(xy)(sh) = z:y((sh)x) = z:y((sh)(xi))$.

Proposition 3.09. Let $(S,X,.,Y,*)$ and $(S',X,.,Y,*)$ be automata and h and h' be the respective mappings from them to egdulk automata. Let h'' be a mapping from S to S'. If (h'',i,i) is an automaton homomorphism, then $h=h''h'$.

Proof: For all s in S and all x and y in X, $y:x(s(h''h')) = y:x((sh'')h') = ((sh'')x)*y = ((sx)h'')*y = ((sx)h'')*(yi) = ((sx)*y)i = (sx)*y = y:x(sh)$.

Proposition 3.10. An automaton is minimal if and only if the homomorphism (h,i,i) of proposition 3.08 is an automaton isomorphism.

Proof: If (h,i,i) is not an isomorphism, then $|Sh|<|S|$. By proposition 3.01, $(S,X,.,Y,*)$ is not minimal.

Assume that $(S,X,.,Y,*)$ is not minimal and let c be the mapping of definition 3.06. Let s and s' be distinct elements of S such that $sc=s'c$. Then h is not one-to-one since $y:x(sh)=(sx)*y=((sc)x)*y=((s'c)x)*y=(s'x)*y=y:x(s'h)$.

Section 3: Machines

Definition 3.11. An Egdulk State is a mapping r from a finitely generated semigroup X onto a finite set of functions from X to a finitely generated semigroup Y with the following properties: The union of the ranges of the functions in the range of r is a set of generators of Y. For all x, y, and z in X, (yz):xr is the product in Y of y:xr and z:(xy)r.

Proposition 3.12. Every state t of an egdulk automaton (T,X,.,Y,*) is an egdulk state which maps X into a set of functions from X into a subsemigroup Y' of Y.
Proof: Let t be any state in T. For all x, y, and z in X, (yz):xt is the product of y:xt and z:(xy)t from definition 3.07. Let A be a finite set of generators of X. Since there are a finite number of distinct functions xt where x is in X and each z:xt can be written as a finite product of a:xt, b:(xa)t, c:(xab)t,..., and g:(xab...f)t where z=abc...fg and a, b, c,..., f, and g are elements of A, the set of all elements a:xt such that a is in A and x in X is a finite set of generators of Y'. Hence Y' is finitely generated.

Proposition 3.13. If r is an egdulk state and T is the set of all functions t such that there exists an x in X for which yt=(xy)r for all y in X, then (T,X,.,Y,*) is an egdulk automaton when $y(r.x)=(xy)r$ and $rx*y=y:xr$ for all x and y in X.

Proof: For each t in T there is an x in X such that for each y in X $yt=(xy)r$. The set of ranges of the functions in the ranges of the mappings in T is equal to the set of ranges of the functions in the range of r. The function t' with the property that $zt'=((xy)z)r$ for all z in X is an element of T. For all z in X, $zt'=((xy)z)r=(x(yz))r=(yz)t$. For all z and w in X, $(zw):yt=(zw):(xy)r$ which is the product in Y of $z:(xy)r$ and $w:((xy)z)r$, and further $z:(xy)r=z:yt$ while $w:((xy)z)r=w:(x(yz))r=w:(yz)t$. Therefore T is a set of mappings for which the transition and output functions of definition 3.07 can be defined.

Definition 3.14. An Egdulk Machine is a machine $(T,X,.,r,Y,*)$ where r and $(T,X,.,Y,*)$ are as in proposition 3.13.

Proposition 3.15. Let $(S,X,.,q,Y,*)$ be a machine. Let r be the mapping from X to the set of functions from X to Y defined by $y:xr=(qx)*y$ for all x and y in X. Let h be defined such that $qh=r$ and $y((qx)h)=(xy)r$ for all x and y in X. The mapping r is an egdulk state and (h,i,i) is a machine homomorphism.

Proof: For any state s in S there is a z in X such that $s=qz$. $y:x(sh) = y:x((qz)h) = y:(zx)r = (qzx)*y = (sx)*y$ for all x and y in X. $(Sh,X,.,Y,*)$ is an automaton and (h,i,i) is an automaton homomorphism by proposition 3.08. By proposition 3.12, r is an egdulk state. By proposition

3.13 and definition 3.14, $(Sh,X,.,r,Y,*)$ is an egdulk machine. Since $qh=r$, (h,i,i) is a machine homomorphism.

Proposition 3.16. Let $(S,X,.,q,Y,*)$ and $(S',X,.,q',Y,*)$ be machines and h and h' be the respective mappings from them to egdulk machines. Let h'' be a mapping from S to S'. If (h,i,i) is a machine homomorphism, then $h=h''h'$.

Proof: Proposition 3.09.

Proposition 3.17. A machine is minimal if and only if the homomorphism (h,i,i) of proposition 3.15 is a machine isomorphism.

Proof: If (h,i,i) is not a machine isomorphism, then $|Sh|<|S|$. By proposition 3.01, $(S,X,.,q,Y,*)$ is not minimal.

Assume that $(S,X,.,q,Y,*)$ is not minimal. Let $(T,X,.,r,Y,*)$ be a machine such that $|T|<|S|$ and $(qx)*y=(rx)*y$ for all x and y in X. Let x and z be elements of X such that qx and qz are distinct while $rx=rz$. $y:x(qh) = (qx)*y = (rx)*y = (rz)*y = (qz)*y = y:z(qh)$ for all y in X. The mapping h is not one-to-one.

Section 4: Semimachines and Semiautomata

The definitions provide that (S,X,.) is the semiautomaton, (S,X,.,q) is the semimachine, and (S,X,.,Y,*) is the automaton of the machine (S,X,.,q,Y,*) while (S,X,.) is the semiautomaton of the automaton (S,X,.,Y,*). The propositions of Chapter 2 can be applied to the machines and automata of Chapter 3 with the help of proposition 3.18.

Proposition 3.18. If (h,i) is a semimachine or semiautomaton isomorphism and (sh)*y=s*y, then (h,i,i) is a machine or automaton isomorphism.

Proof: (s*x)i=s*x=(sh)*x=(sh)*(xi). The rest follows from the definitions.

Each machine defines an input right congruence E(r) and an input congruence E(S). A machine is isomorphic to a machine with a right congruence semimachine for its semimachine by proposition 2.11 and is a homomorphic image of a machine which has as its semimachine the right congruence semimachine of the input congruence of the former machine by proposition 2.13.

Proposition 3.19. The input right congruence of any machine refines that of its egdulk machine.

Proof: (h,i,i) of proposition 3.15 is a machine homomorphism. (h,i) is a semimachine homomorphism. By proposition 2.14, the former input right congruence refines the latter.

The semiautomaton of each automaton is isomorphic to a semiautomaton of the type described in proposition 2.32. Using proposition 3.18, a canonical form with functions into the integers for states can be defined.

CHAPTER 4: MULTIPROGRAMMING

Section 1: General Discussion

Definition 4.01. A Multiprogramming Machine is a machine $(S',X',.,p,Y',*)$ with submachine $(S,X,.,p,Y,*)$ such that there exist machines $(V,A,.,q,B,*)$ and $(W,C,.,r,D,*)$ where A and C are contained in X and B and D in Y and semigroup homomorphisms $j:X\text{--}A$, $k:Y\text{--}B$, $j':X\text{--}C$, and $k':Y\text{--}D$ such that $aj=a$ and $aj'=e$ for all a in A, $bk=b$ and $bk'=e$ for all b in B, $cj=e$ and $cj'=c$ for all c in C, $dk=e$ and $dk'=d$ for all d in D, and such that $(q(xj))*(yj)=((px)*y)k$ and $(r(xj'))*(yj')=((px)*y)k'$ for all x and y in X. The first machine is said to simulate the last two.

Definition 4.02. A Multiprogramming Automaton is an automaton $(S',X',.,Y',*)$ with subautomaton $(S,X,.,Y,*)$ such that there exist automata $(V,A,.,B,*)$ and $(W,C,.,D,*)$ where A and C are contained in X and B and D in Y and semigroup homomorphisms $j:X\text{--}A$, $k:Y\text{--}B$, $j':X\text{--}C$, and $k':Y\text{--}D$ and relations G and H in SxV and SxW such that $aj=a$ and $aj'=e$ for all a in A, $bk=b$ and $bk'=e$ for all b in B, $cj=e$ and $cj'=c$ for all c in C, $dk=e$ and $dk'=d$ for all d in D, there exists an s in S for each (v,w) in VxW such that (s,v) is in G, and (s,w) is in H, $((sx)*y)k=(v(xj))*(yj)$ for all x and y in X and all (s,v) in G, and

$((sx)*y)k'=(w(xj'))*(yj')$ for all x and y in X and all (s,w) in H. The first automaton is said to simulate the last two.

In less precise language, a multiprogramming automaton is one which can simulate two other automata operating independently, starting at any given state and each receiving its inputs while disregarding inputs to the other. One or both of these may in turn be a multiprogramming automaton so that one automaton may be capable of simulating any finite number of automata.

There may be states in S which are not related by G or H to any states of V or W. There may be inputs in X' or outputs in Y' that are not in the semigroups generated by AUC and BUD. In this way, a multiprogramming automaton may be able to do more than simulate two automata operating independently. In particular, there may be control inputs which may interact with the elements of A and C.

This section treats the case where $S=S'$, $X=X'$, and $Y=Y'$. The addition of extra states does not change the properties of the other states. Adding elements to the input semigroup may affect several of the properties of the system. Additional output elements may be included with added input elements. The hypotheses will include any needed references to these semigroups.

<u>Proposition 4.03</u>. Let $(V,A,.,q,B,*)$ and $(W,C,.,r,D,*)$ be machines. Let X be the free product monoid of A and C, that is, every element of X can be

written uniquely as a product of elements of $A \cup C$ no two adjacent terms in the product being both from A or both from C where the identity element of X is in both A and C. Let Y be the free product monoid of B and D. The machine $(V\times W, X, ., (q,r), Y, *)$, the internal product machine of the first two, is a multiprogramming machine which simulates them.

Proof: Let j and j' be the mappings from X into A and C defined such that $aj=a$, $cj=e$, $(xc)j=(cx)j=xj$, $(ax)j=(aj)(xj)$, $(xa)j=(xj)(aj)$, $aj'=e$, $cj'=c$, $(cx)j'=(cj')(xj')$, $(xc)j'=(xj')(cj')$, and $(xa)j'=(ax)j'=xj'$, for all a in A, c in C, and x in X. Both j and j' are clearly semigroup homomorphisms. Since $(v,w)ax=(va,w)x=(v(aj),w)x$ and $(v,w)cx=(v,wc)x=(v,w(cj'))x$ for all (v,w) in $V\times W$, a in A, c in C, and x in X, $(q,r)x=(q(xj),r(xj'))$ and $(v,w)x=(v(xj),w(xj'))$ for all x in X and (v,w) in $V\times W$.

Let k and k' be the mappings from Y into B and D defined such that $bk=b$, $dk=e$, $(yd)k=(dy)k=yk$, $(by)k=(bk)(yk)$, $(yb)k=(yk)(bk)$, $bk'=e$, $dk'=d$, $(dy)k'=(dk')(yk')$, $(yd)k'=(yk')(dk')$, and $(by)k'=(yb)k'=yk'$, for all b in B, d in D, and x in X. Both k and k' are clearly semigroup homomorphisms. Since $(v,w)*(acx)$ $=$ $((v,w)*a)(((v,w)a)*c)(((v,w)ac)*x)$ $=$ $((v,w)*a)((va,w)*c)((va,wc)*x) = (v*a)(w*c)((va,wc)*x)$ and similarly $(v,w)*(cax) = (w*c)(v*a)((va,wc)*x)$ while $(v*a)k=v*a$, $(v*a)k'=e=(w*c)k$, and $(w*c)k'=w*c$ for all (v,w)

in $V \times W$, a in A, c in C, and x in X, it follows that $(((q,r)x)*y)k = ((q(xj),r(xj'))*y)k = (q(xj))*(yj)$ while $(((q,r)x)*y)k' = (r(xj'))*(yj')$.

Proposition 4.04. Let $(V,A,.,B,*)$ and $(W,C,.,D,*)$ be automata. Let X be the free product monoid of A and C and Y the free product monoid of B and D. The automaton $(V \times W,X,.,Y,*)$, the internal product automaton of the first two, is a multiprogramming automaton which simulates them.

Proof: Let G be the set of all $((v,w),v)$ and H be the set of all $((v,w),w)$ with (v,w) in $V \times W$. Define the semigroup homomorphisms j, j', k, and k' as in the proof of proposition 4.03. If (s,v) is in G, then there is a w in W such that $s=(v,w)$. Hence, by proposition 4.03, $((sx)*y)k = (((v,w)x)*y)k = (v(xj))*(yj)$ in the submachine with start state s. Similarly, there is a v in V for each (s,w) in H such that $s=(v,w)$ and $((sx)*y)k' = (((v,w)*y)k' = (w(xj'))*(yj')$. The output function on a submachine is the restriction of the output function of the automaton to the states in the submachine.

Proposition 4.05. Let $(V,A,.,q,B,*)$ and $(W,C,.,r,D,*)$ be machines such that A and C are disjoint subsemigroups of a semigroup X and B and D are disjoint subsemigroups of a semigroup Y. If there is a machine $(S',X,.,p,Y,*)$ that has fewer states than $V \times W$ and can simulate the first two machines, then at least one of the first two is not minimal.

Proof: Let j, j', k, and k' be as in definition 4.01. Let h be the mapping from V to egdulk states and h' be the mapping from W to egdulk states such that (h,i,i) and (h',i,i) are machine homomorphisms to egdulk machines of the type in proposition 3.15. Since S' has fewer states than VxW, there is at least one pair of products ac and a'c' in X such that pac=pa'c' in S' while (qa,rc)≠(qa',rc') in VxW. If qa≠qa', then h is not one-to-one, since, for any x and y in X, (yj):(xj)((qa)h) = (yj):(xj)((q((ac)j)h) = (q((ac)j)(xj))*(yj) = (q((acx)j))*(yj) = ((pacx)*y)k = ((pa'c'x)*y)k = (q((a'c'x)j))*(yj) = (q((a'c')j(xj))*(yj) = (yj):(xj)((q((a'c')j)h) = (yj):(xj)((qa')h). If rc≠rc', then h' is not one-to-one, since it can be shown in a similar manner that (rc)h'=(rc')h'. Hence, one of the two homomorphisms is not an isomorphism and, by proposition 3.16, one of the two machines is not minimal.

<u>Proposition</u> <u>4.06</u>. Let (V,A,.,B,*) and (W,C,.,D,*) be automata such that A and C are disjoint subsemigroups of a semigroup X and B and D are disjoint subsemigroups of a semigroup Y. If there is an automaton (S',X,.,Y,*) that has fewer states than VxW and can simulate the first two automata, then at least one of the first two is not minimal.

Proof: Let j, j', k, k', G, and H be as in definition 4.02. Let h be the mapping from V to egdulk states and h' be the mapping from W to egdulk states such that (h,i,i) and (h',i,i) are automaton homomorphisms of the type in proposition 3.08. Since S has fewer states than $V\times W$, there is at least one distinct pair of elements of $V\times W$, (v,w) and (v',w'), such that there is one element s in S for which both (s,v) and (s,v') are in G and both (s,w) and (s,w') are in H. If $v\neq v'$, then h is not one-to-one, since, for any x and y in X, $(yj):(xj)(vh) = (v(xj))*(yj) = ((sx)*y)k = (v'(xj))*(yj) = (yj):(xj)(v'h)$. If $w\neq w'$, then h' is not one-to-one, since, for any x and y in X, $(yj'):(xj')(wh') = (w(xj'))*(yj') = ((sx)*y)k' = (w'(xj'))*)yj')$. Hence, one of the two homomorphisms is not an isomorphism and, by proposition 3.10, one of the two automata is not minimal.

Propositions 4.05 and 4.06 say that no multiprogramming system is more efficient than the product of efficient systems. While propositions 4.03 and 4.04 showed the existence of a multiprogramming system simulating any two given systems as an internal product when its semigroups are free products of those of the simulated systems. the following propositions examine some of the properties of systems with other semigroups.

<u>Proposition 4.07</u>. If the automata $(S,X,.,Y,*)$, $(V,A,.,B,*)$, and $(W,C,.,D,*)$ as defined in definition 4.02 are egdulk automata having the property that $s*a$ is in B and $s*c$ is in D for each s in S, a in A, and c in C where X is generated by $A\cup C$ and $P(v)$ and $P(w)$ are, respectively, the sets of all s such that (s,v) is in G and (s,u) is in H for some u in W and such that (s,w) is in H and (s,u') is in G for some u' in V, then for each w in W $(P(w),A,.,B,*)$

is isomorphic to $(V,A,.,B,*)$ while for each v in V $(P(v),C,.,D,*)$ is isomorphic to $(W,C,.,D,*)$.

Proof: If s is in the intersection of $P(v)$ and $P(v')$ where v and v' are in V, then $v=v'$ since for all a and a' in A $a':av = va*a' = (v(aj))*a'j) = (sa*a')k = (v'(aj))*(a'j) = v'a*a' = a':av'$. If s is in the intersection of $P(w)$ and $P(w')$ where w and w' are in W, then $w=w'$. There is at least one s in the intersection of each $P(v)$ and each $P(w)$. Assume that there is a (v,w) in VxW such that s and t are in the intersection of $P(v)$ and $P(w)$. If s and t are distinct, then there is some x and some y in X such that $y:xs \neq y:xt$, and since y is in the semigroup generated by AUC, there are y' and y'' in X and z in AUC such that $y=y'zy''$ and $z:(xy')s \neq z:(xy')t$. If z is in A, then $z:(xy')s = (z:(xy')s)k = (zj):((xy')j)v = (z:(xy')t)k = z:(xy')t$. Similarly, if z is in C, then $z:(xy')s = z:(xy')t$. Hence, s and t cannot be distinct. For any w in W, define the mapping h from $P(w)$ to V by restricting G to $P(w)xV$. For any v in V, define the mapping h' by restricting H to $P(v)xW$. The triples of mappings (h,j,k) and (h',j',k') are automaton homomorphisms. The triples of mappings $(h,j|A,k|B)$ and $(h',j'|C,k'|D)$ are automaton homomorphisms which, since they are one-to-one, are isomorphisms.

Proposition 4.08. If the machines $(S,X,.,p,Y,*)$, $(V,A,.,q,B,*)$, and $(W,C,.,r,D,*)$ as defined in definition 4.01 are egdulk machines having the property that $s*a$ is in B and $s*c$ is in D for each s in S, a in A, and c in C where

X is generated by $A \cup C$, then for each c in C the submachine $(T,A,.,pc,B,*)$ of $(S,X,.,p,Y,*)$ is isomorphic to $(V,A,.,q,B,*)$ while for each a in A the submachine $(T',C,.,pa,D,*)$ of $(S,X,.,p,Y,*)$ is isomorphic to $(W,C,.,r,D,*)$.

Proof: The relations G and H of definition 4.02 and subsets $P(v)$ and $P(w)$ of proposition 4.07 can be defined. For each w in W $(P(w),A,.,B,*)$ is automaton isomorphic to $(V,A,.,B,*)$ and for each v in V $(P(v),C,.,D,*)$ is automaton isomorphic to $(W,C,.,D,*)$ by proposition 4.07. Let c be some element of C. Since $(rc(xj'))*(yj') = (r(cj')(xj'))*(yj') = (r((cx)j'))*(yj') = ((pcx)*y)k'$, (pc,rc) is in H and pc is in $P(rc)$. Since $(rc(xj'))*(yj') = (r(cj')(aj')(xj'))*(yj') = (r((cax)j'))*(yj') = ((pcax)*y)k'$, pca is in $P(rc)$ for all a in A. Since $(P(rc),A,.,B,*)$ is automaton isomorphic to $(V,A,.,B,*)$ which is the automaton of a machine, any element of $P(rc)$ can be expressed as pca. In particular, since $(q(xj))*(yj) = (q(cj)(xj))*(yj) = (q((cx)j))*(yj) = (qcx*y)k$, $(pc)h=q$ and each automaton isomorphism is a machine isomorphism. Similarly, each automaton isomorphism $(h',j'|C,k'|D)$ as defined using proposition 4.07 is a machine isomorphism with $(pa)h'=r$.

Although the previous two propositions may seem obvious and the conditions on the output function unnecessary, there are egdulk automata which have their state spaces contained in the first projections of both of the relations G and H and yet have more states than VxW. For example, for some s and t in S and a in A, the output mapping might be defined so that s*a=d'bd and t*a=b where b is in B and d' and d are mutually inverse in D.

Proposition 4.09. Let the automata, mappings, and relations be as defined in definition 4.02. If S is in the first projections of both G and H, then there exist automata (V',A,.,B,*), (W',C,.,D,*), and (V'xW',X,.,Z,*) and semigroup homomorphisms f":Y--Z, f:Z--B, and f':Z--D and mappings h:V--V', h':W--W', h":S--V'xW', g:V'xW'--V', and g':V'xW'--W' such that yf"f=yk and yf"f'=yk' for all y in Y, (bd)f"=(db)f" for all b in B and d in D, (v',w')g=v'and (v',w')g'=w' for all (v',w') in V'xW', sh"=(vh,wh') for all (s,v) in G and (s,w) in H, and the triples of mappings (h,i,i):(V,A,.,B,*)--(V',A,.,B,*), (g,j,f):(V'xW',X,.,Z,*)--(V',A,.,B,*), (h",i,f"):(S,X,.,Y,*)--(V'xW',X,.,Z,*), (g',j',f'):(V'xW',X,.,Z,*)--(W',C,.,D,*), and (h',i,i):(W,C,.,D,*)--(W',C,.,D,*) are all automaton homomorphisms.

$$\begin{tikzcd}
 & (S,X,.,Y,*) \arrow[d, "(h'',i,f'')"] & \\
(V,A,.,B,*) \arrow[d, "(h,i,i)"] & (V'xW',X,.,Z,*) \arrow[dl, "(g,j,f)"'] \arrow[dr, "(g',j',f')"] & (W,C,.,D,*) \arrow[d, "(h',i,i)"'] \\
(V',A,.,B,*) & & (W',C,.,D,*)
\end{tikzcd}$$

Proof: Let E be the intersection of K(k) and K(k'), Z=X/E, and f" be the natural homomorphism from Y to Z. For all b in B and d in D, since (bd)k=(bk)(dk)=be=eb=(dk)(bk)=(db)k and (bd)k'=ed=de=(db)k', it follows that (bd,db) is in E, (bd)f"=(db)f", and E=K(f"). Since K(f") is a subset of K(k) there is a unique homomorphism f such that yk=yf"f for all y in Y (CP). There is likewise a unique homomorphism f' such that yk'=yf"f' for all y in Y.

Let (V',A,.,B,*) and (W',C,.,D,*) be egdulk automata and (h,i,i) and (h',i,i) the respective automaton homomorphisms from (V,A,.,Y,*) and (W,C,.,D,*) as defined in proposition 3.08. Let g be the projection from V'xW' to V' and g' the projection to W'. If (s,v) and (s,v') are both in G, then vh=v'h because for all a and a' in A a':a(vh)=va*a'=((sa)*a')k=v'a*a'=a':a(v'h). Similarly wh'=w'h' if (s,w) and (s,w') are both in H. The mapping h" is defined by sh"=(vh,wh') where (s,v) is in G and (s,w) in H.

For the transition and output functions of (V'xW',X,.,Z,*) define (vh,wh')x=((vh)(xj),(wh')(xj')) and (vh,wh')*x=((vh)*(xj))((wh')*(xj')). That (vh,wh')*(xy) = ((vh)*((xy)j))((wh')*((xy)j')) = ((vh)*(xj))((vh)(xj)*(yj))((wh')*(xj'))((wh')(xj')*(yj')) = ((vh)*(xj))((wh')*(xj'))((vh)(xj)*(yj))((wh')(xj')*(yj')) = ((vh,wh')*x)((vh,wh')x*y) for all x and y in X, (vh,wh')*a = ((vh)*(aj))((wh')*(aj')) = ((vh)*a)e = (vh)*a for all a in A, and (vh,wh')*c = (wh')*c for all c in C whenever (vh,wh') is in V'xW' shows that (V'xW',X,.,Z,*) is the internal product automaton of (V',A,.,B,*) and (W',C,.,D,*). Since j and g are semigroup homomorphisms and $((vh,wh')*x)f = (((vh)*(xj))((wh'*(xj')))f = (((vh)*(xj))f)(((wh')*(xj'))f) = ((vh)*(xj))e = ((vh,wh')g)*(xj)$, (g,j,f) is an automaton homomorphism. The triple (g',j',f') is also an automaton homomorphism. Since $(sh'')*x = (vh,wh')*x = ((vh)*(xj))((wh')*(xj')) = (s*x)f''$ for all (s,v) in G, (s,w) in H, and x in X, (h'',i,f'') is an automaton homomorphism.

<u>Proposition</u> <u>4.10</u>. Let the machines and mappings be defined as in definition 4.01. There exist machines $(V',A,.,qh,B,*)$, $(W',C,.,rh,D,*)$, and $(V'xW',X,.,(qh,rh),Z,*)$ and semigroup homomorphisms f'', f, and f', and mappings h, h', h'', g, and g' as in proposition 4.09 such that $yf''f=yk$ and $yf''f'=yk'$ for all y in Y, $(bd)f''=(db)f''$ for all b in B and d in D, $(v',w)g=v'$ and $(v',w')g'=w'$ for all (v',w') in $V'xW'$, $(px)h''=((q(xj))h,(r(xj'))h)$ for all x in X, and the triples of mappings $(h,i,i):(V,A,.,q,B,*)\text{--}(V',A,.,qh,B,*)$, $(g,j,f):(V'xW',X,.,(qh,rh'),Z,*)\text{--}(V',A,.,qh,B,*)$, $(h'',i,f''):(S,X,.,p,Y,*)\text{--}(S,X,.,(qh,rh'),Z,*)$, $(g',j',f'):(V'xW',X,.,(qh,rh'),Z,*)\text{--}(W',C,.,rh',D,*)$, and $(h',i,i):(W,C,.,r,D,*)\text{--}(W',C,.,rh',D,*)$ are all machine homomorphisms.

$$\begin{tikzcd}
 & (S,X,.,p,Y,*) \arrow[dd, "{(h'',i,f'')}"] & \\
(V,A,.,q,B,*) \arrow[dd, "{(h,i,i)}"] & & (W,C,.,r,D,*) \arrow[dd, "{(h',i,i)}"'] \\
 & (V'xW',X,.,(rh,qh'),D,*) \arrow[dl, "{(g,j,f)}"] \arrow[dr, "{(g',j',f')}"'] & \\
(V',A,.,ph,B,*) & & (W',C,.,qh',D,*)
\end{tikzcd}$$

Proof: Let G be the set of all (px,q(xj)) and H the set of all (px,r(xj')) such that x is in X. S is the first projection of both G and H and (S,X,.,Y,*) is a multiprogramming automaton simulating (V,A,.,B,*) and (W,C,.,D,*). The existence of the automata and automaton homomorphisms follows from proposition 4.09. Since (p,q) is in G and (p,r) is in H, ph"=(vh,wh'). By proposition 3.15, (h,i,i) and (h',i,i) are machine homomorphisms. Since (ph")g=(qh,rh')g=qh and (ph")g'=(qh,rh')g'=rh', (h,i,i) and (h',i,i) are machine homomorphisms.

Proposition 4.11. If (S',X',.,Y',*) is a multiprogramming automaton simulating (V,A,.,B,*) and (W,C,.,D,*) and is a subautomaton of (S",X",.,Y",*), then (S",X",.,Y",*) simulates (V,A,.,B,*) and (W,C,.,D,*).

Proof: The automaton (S,X,.,Y,*) of the definition is also a subatomaton of (S",X",.,Y",*). The mappings from S, X, and Y and the relations G and H serve to show that any automaton with this subautomaton simulates (V,A,.,C,*) and (W,C,.,D,*).

Proposition 4.12. If (S',X',.,p,Y',*) is a multiprogramming machine simulating (V,A,.,q,B,*) and (W,C,.,r,D,*) and is a submachine of (S",X",.,p,Y",*), then (S",X",.,p,Y",*) simulates (V,A,.,q,B,*) and (W,C,.,r,D,*).

Proof: The machine (S,X,.,p,Y,*) of the definition is also a submachine of (S",X",.,p,Y",*) with the same start state. The mappings from S, X, and Y serve to show that any machine with the same start state which has this machine as a submachine simulates (V,A,.,q,B,*) and (W,C,.,r,C,*).

<u>Proposition</u> <u>4.13</u>. If (V',A',.,q,B',*) and (W',C',.,r,D',*) are respectively submachines of (V,A,.,q,B,*) and (W,C,.,r,D.*) and the latter pair of machines is simulated by (S',X',.,p,Y',*), then the former pair of machines is also simulated by the same multiprogramming machine.

Proof: Let X" be the subsemigroup of X' generated by the union of A' and C'. Let (S",X",.,p,Y",*) be a submachine of (S',X',.,p,Y',*). Since X" is a subsemigroup of X, (S",X",.,p,Y",*) is a submachine of the (S,X,.,p,Y,*) defined in definition 4.01. The restrictions of j and j' to X" and of k and k' to Y" are the mappings which show that (S',X',.,p,Y',*) simulates (V',A',.,q,B',*) and (W',C',.,r,D',*).

<u>Proposition</u> <u>4.14</u>. If (V',A',.,B',*) and (W',C',.,D',*) are respectively subautomata of (V,A,.,B,*) and (W,C,.,D,*) and the latter pair of automata is simulated by (S',X',.,Y',*), then the former pair of automata is also simulated by the same multiprogramming automaton.

Proof: Let X" be the subsemigroup of X' generated by the union of A' and C'. Since X" is a subsemigroup of X, there is a subsemigroup Y" of Y such that (S,X",.,Y",*) is a subautomaton of (S,X,.,Y,*) as defined in definition 4.02. The relations G and H and the restrictions of j and j' to X" and of k and k' to Y" show that (S',X',.,Y',*) simulates

(V',A',.,B',*) and (W',C',.,D',*).

Proposition 4.15. Any machine is trivially a multiprogramming machine.

Proof: Let (S,X,.,p,Y,*)=(V,A,.,q,B,*) and let (W,C,.,r,*) be a single state machine with input and output semigroups having only single elements. j and k are identity mappings on X and Y while j' and k' map all elements of each to its identity element.

Proposition 4.16. Any automaton is trivially a multiprogramming automaton.

Proof: Let (S,X,.,Y,*)=(V,A,.,B,*) and let (W,C,.,D,*) be a single state machine with input and output semigroups having only single elements. j and k are identity mappings on X and Y while j' and k' map all elements of each to its identity element. G is the identity relation on S while H is the set of all (s,w) such that s is in S and w is the element of W.

Definition 4.17. A Trivial Monoid is a monoid with a single element.

Definition 4.18. A Trivial Machine, Automaton, Semimachine, or Semiautomaton is one with a trivial input semigroup.

As a result of proposition 4.13, the investigation of the existence of a pair of nontrivial machines which a given machine (S',X',.,p,Y',*) can simulate can be limited to machines with cyclic subsemigroups of X' for input. This investigation can be further limited to nontrivial cyclic monoids without proper minimal nontrivial submonoid.

Proposition 4.19. Every nontrivial monoid contains a submonoid where the identity of the submonoid is that of the monoid of one of the following types: a 2-element semilattice, a group of prime order, or an infinite order cyclic monoid. Each of these is nontrivial but contains no nontrivial proper submonoids.

Proof: Any element other than the identity of a monoid generates a cyclic submonoid. A cyclic monoid contains an identity element and all powers of some element b. If the monoid is infinite, any submonoid is either the trivial one or a monoid generated by some power of b. Neither type is both minimal and nontrivial. If the monoid is finite, some power of b is an idempotent. If this idempotent is not the identity, the monoid consisting of the idempotent power of b and the identity is a 2-element semilattice every submonoid of which is trivial. If this idempotent is the identity, let the n-th power of b be the identity. If n is prime, there is only one submonoid and it is trivial. If n is not prime and n/m is prime, the m-th power of b generates a nontrivial submonoid of order n/m.

Proposition 4.20. If (S',X',.,Y',*) is a multiprogramming automaton simulating two nontrivial automata or if (S',X',.,p,Y') is a multiprogramming machine simulating two nontrivial machines, then there are in X' two elements a and c which generate monoids of prime or infinite order such that the only power of a which is a power of c is the identity.

Proof: If (S',X',.,Y',*) simulates nontrivial automata (V,A,.,B,*) and (W,C,.,D,*), A and C are nontrivial and there are, by proposition 4.19, an a in A and a c in C which generate submonoids of prime or infinite order. If x is a power of both a and c, then xj=x because it is a power of a while x=xj=e because it is a power of c.

Proposition 4.21. If (S',X',.,Y',*) is an automaton such that there are elements a and c in X' such that the monoid generated by each is either infinite or a group of prime order or a semilattice of order two and the natural homomorphism from the monoid X generated by a and c to X/E where E is the congruence on X generated by (ac,ca) is one-to-one when restricted to the setwise product of the cyclic monoid generated by a and that generated by c, then there are nontrivial automata (V,A,.,B,*) and (W,C,.,D,*) such that (S',X',.,Y',*) simulates them.

Proof: Let (S,X,.,Y,*) be any subautomaton such that X is generated by a and c. Let A be the submonoid generated by a and C that generated by c. Let f be the natural homomorphism from X to X/E. For any x in X there is a unique a' in A and a unique c' in C such that xf=(a'c')f. Define j and j' such that xj=a' and xj'=c' for each x in X. These mappings are endomorphisms since for any x and y in X (xf)(yf) = ((xj)(xj'))f((yj)(yj'))f =

$((xj)(xj')(yj)(yj'))f = ((xj)(yj)(xj')(yj'))f = ((xy)j(xy)j')f = (xy)f$. Let k and k' be the constant mappings from Y to the identity of Y. Let the machines $(S,A,.,B,\#)$ and $(S,C,.,D,\#)$ be defined with transition functions agreeing with $(S,X,.,Y,*)$ and all output the identity. Let G and H be the identity relation on S. For all x and y in X and all (s,s) in G $((sx)*y)k=(s(xj))\#(yj)$. For all x and y in X and all (s,s) in H $((sx)*y)k'=(s(xj'))\#(yj')$.

<u>Proposition</u> <u>4.22</u>. If $(S',X',.,p,Y',*)$ is a machine such that there are elements a and c in X' such that the monoid generated by each is either infinite or a group of prime order or a semilattice of order two and the natural homomorphism from the monoid X generated by a and c to X/E where E is the congruence on X generated by (ac,ca) is one-to-one when restricted to the setwise product of the cyclic monoid generated by a and that generated by c, then there are nontrivial machines $(V,A,.,q,B,*)$ and $(W,C,.,r,D,*)$ such that $(S',X',.,p,Y',*)$ simulates them.

Proof: By proposition 4.21, the automaton of this machine simulates two nontrivial machines. By proposition 4.14, it simulates the automaton of any submachine of them. Any submachine of a nontrivial automaton which has the same input monoid is likewise nontrivial. In particular, the automaton of any submachine starting with a state q such that (p,q) is in G or state r such that (p,r) is in H is simulated as automaton by the automaton $(S',X',.,Y',*)$.

Hence, $((px)*y)k=(q(xj))*(yj)$ and $((px)*y)k'=(r(xj'))*(yj')$ and they are simulated as machines.

Section 2: Free Monoids of Input and Output

Definition 4.23. The relation C(R) is defined for any relation R on the generators of a finitely generated free monoid X to be the congruence relation generated by the set of all pairs (ab,ba) such that a and b are generators of X but neither (a,b) nor (b,a) is in R.

Proposition 4.24. A necessary and sufficient condition for C(Q) to be a subset of C(R) is that R be a subset of the union of Q, the inverse relation of Q, and the identity relation.

Proof: Assume that for every (a,b) in R either a=b, (a,b) is in Q, or (b,a) is in Q. Let x and y be two elements of the finitely generated free monoid X such that (x,y) is in C(Q). Either there is a finite sequence of words x',x",...,y",y' such that x differs from x', y' differs from y, and each word in the sequence differs from the next only in the transposition of two distinct generators which are not related by Q or the inverse of Q or x differs from y only in the same manner or x=y. In the first case each of the pairs (x,x'), (x',x"), ..., (y",y'), and (y',y) is in the congruence relation C(R) since the transposed pair of generators was not in either Q or its inverse and hence not in R. In the second case (x,y) is similarly in C(R).

In the last case (x,y) is in C(R) because it is an equivalence relation.

Assume now that there is a pair (a,b) of distinct generators which is in R but in neither Q nor its inverse. The pair of words (ab,ba) is in C(Q). Since there is no word which differs from ab only in the transposition of two generators not related by R, (ab,ab) is the only pair in C(R) with ab for an entry. Since ab≠ba, (ab,ba) is not in C(R).

Proposition 4.25. If R and Q are equivalence relations, then a necessary and sufficient condition for C(Q) to be a subset of C(R) is that R be a subset of Q.

Proof: This is a corollary of proposition 4.24 since Q is its own inverse and contains the identity relation.

Proposition 4.26. If X is the free monoid generated by the finite set X', then for any R in X'xX' $C(X'xX')\leq C(R)\leq C(I)=C(\emptyset)$ where I is the identity relation and ∅ the empty set.

Proof: Since $IU\emptyset=IUI\leq RUI\leq X'xX'$, this is another corollary of proposition 4.24.

Proposition 4.27. If R is an equivalence relation on the generators of a finitely generated free monoid X and P(1),P(2),...,P(m) are the equivalence classes of P, then for each x in X there are elements a,b,...,d respectively of the free monoids generated by P(1),P(2),...,P(m) such that (x,ab...d) is in C(R).

Proof: The proof is trivial for words of length 0 or 1 since $e=ee\ldots e$ and for each generator z $z=e\ldots eze\ldots e$. Assume for any m greater than 1 that the proposition is true for all words of length less than m. Let x be any word of length m. Since x is the product of two words, z' of length $m-1$ and z of length 1, $(z',ab\ldots d)$ and $(x,ab\ldots dz)$ are in $C(R)$. If z is in the monoid generated by $P(m)$, so is dz. Otherwise if $ab\ldots d=y'y$ where z is in $P(m')$ y' is a product of elements from $P(1)$ through $P(m'-1)$ and y a product of elements from $P(m'+1)$ through $P(m)$ (yz,zy) is in $C(R)$ because $y=e$ or the sequence of words with z transposed with each of the generators of y has each element $C(R)$ related to yz. Finally $(x,ab\ldots cz\ldots d)$ where cz is in the monoid generated by $P(m')$.

Proposition 4.28. If the semigroups A and C of definition 4.01 or 4.02 are finitely generated free monoids and X is their free product and R is the equivalence relation on the generators of X with the generators of A in one class and those of C in the other, $C(R)$ is the intersection of the kernels $K(j)$ and $K(j')$. If the output semigroups B, D, and Y are similarly related and Q is the analogous relation on the generators of Y, $C(Q)$ is the intersection of the kernels $K(k)$ and $K(k')$.

Proof: Each kernel of a homomorphism is a congruence relation. The intersection of congruence relations is a congruence relation. For each a in A and c in C $(ac)j=(aj)(cj)=ae=ea=(cj)(aj)=(ca)j$ and

$(ac)j'=ec=ce=(ca)j'$. Since each pair (ac,ca) and (ca,ac) where a is in A and c in C is in the intersection of the kernels $K(j)$ and $K(j')$ and $C(R)$ is the congruence generated by such pairs, $C(R)$ is a subset of the intersection of kernels. For each x in X there are a in A and c in C such that $a=aj=xj$ and $c=cj'=xj'$. If (x,y) is in the intersection of the kernels, $a=yj$ and $c=yj'$. From proposition 4.14, there are a' in A and c' in C such that $(a'c',x)$ is in $C(R)$. Since $C(R)$ is in the intersection of kernels, $a'=(a'c')j=xj=a$ and $c'=(a'c')j'=xj'=c$ and (x,ac) is in $C(R)$. Similarly (y,ac) is in $C(R)$. Hence the intersection of $K(j)$ and $K(j')$ is in $C(R)$. The intersection of $K(k)$ and $K(k')$ is shown to be equal to $C(Q)$ by a similar argument.

<u>Proposition 4.29</u>. If the monoids A, B, C, D, X, and Y and the mappings j, j', k, and k' are defined as in proposition 4.28, then for all x in X and y in Y $xj'=e$ only if $xj=x$, $xj=e$ only if $xj'=x$, $yk'=e$ only if $yk=y$, and $yk=e$ only if $yk'=y$.

Proof: By proposition 4.27, for each x in X there are a in A and c in C such that (x,ac) is in $C(R)$ where R is defined as in proposition 4.14. Since $C(R)$ is in $K(j')$, if $xj'=e$, then $e=(ac)j'=(aj')(cj')=e(cj')=c$. Since $ac=ae$ is $C(R)$ related only to itself, $x=ae=a$ and $xj=aj=x$. The proof of each of the other parts of this proposition is similar.

Proposition 4.30. If (S,X,.,Y,*) is an automaton with Y a finitely generated free monoid, then s*e=e for all s in S.

Proof: For any s in S, s*e=s*(ee)=(s*e)(se*e)=(s*e)(s*e). Since the only idempotent in a free monoid is the identity, s*e=e.

Proposition 4.31. If the machines (S,X,.,p,Y,*), (V,A,.,q,B,*), and (W,C,.,r,D,*) as defined in definition 4.01 are egdulk machines and A, B, C, D, X, and Y are free monoids where the set of generators of X is the union of the sets of generators of A and C and the set of generators of Y is the union of the sets of generators of B and D, then for each c in C the submachine (T,A,.,pc,B,*) of (S,X,.,p,Y,*) is isomorphic to (V,A,.,q,B,*) while for each a in A the submachine (T',C,.,pa,D,*) of (S,X,.,r,Y,*) is isomorphic to (W,C,.,r,D,*).

Proof: Since (se*a)k'=(s(ej'))*(aj')=se*e=e by proposition 4.30, ve*a=(se*a)k=se*a by proposition 4.29 for all a in A. Similarly we*c=se*c for all c in C. Proposition 4.08 completes the proof.

Proposition 4.32. If the automata (S,X,.,Y,*), (V,A,.,B,*), and (W,C,.,D,*) as defined in definition 4.02 are egdulk automata and A, B, C, D, X, and Y are free monoids where the set of generators of X is the union of the sets of generators of A and C and the set of generators of Y is the union of the sets of generators of B and D and P(v) and P(w) are defined as in proposition 4.07, then for each w in W (P(w),A,.,B,*) is isomorphic to (V,A,.,B,*) while for each v in V (P(v),C,.,D,*) is isomorphic to (W,C,.,D,*).

Proof: Since (se*a)k'=se*e=e by proposition 4.30 and ve*a=(se*a)k=se*a by proposition 4.29, the result is immediate from proposition 4.07.

<u>Proposition</u> <u>4.33</u>. Any machine (S',X',.,p,Y',*) with X' a noncyclic free monoid is, nontrivially, a multiprogramming machine.

Proof: Let a and c be any two distinct generators of X'. The submonoid X generated by a and c is the free product monoid of the cyclic submonoids generated by a and by c. Let E be C(I). C(I) is the congruence relation generated by (ac,ca). If a' and a" are powers of a and c' and c" are powers of c and (a'c',a"c") is in C(I), then a'=a" and c'=c". Hence, the natural homomorphism onto X/E is one-to-one from the setwise product of the cyclic monoids generated by a and by c in that order. By proposition 4.22, there are non-trivial machines (V,A,.,q,B,*) and (W,C,.,r,D,*) such that (S',X',.,p,Y',*) simulates them.

<u>Proposition</u> <u>4.34</u>. Any automaton (S',X',.,Y',*) with X' a noncyclic free monoid is, nontrivially, a multiprogramming automaton.

Proof: Once more let a and c be distinct generators of X' and E be C(I) where I is the identity relation on the set with elements a and b. By proposition 4.21 there are nontrivial automata (V,A,.,B,*) and (W,C,.,D,*) such that

$(S',X',.,Y',*)$ simulates them.

Proposition 4.35. Let $(S',X',.,p,Y',*)$ be a machine where X' and Y' are both noncyclic free monoids and each $px*x'$ such that x is an element of X' and x' is a generator of X' is one of the generators of Y'. The existence of generators a and c of X' such that $px*a \neq px'*c$ for all x and x' in the submonoid generated by a and c is a necessary and sufficient condition for the existence of machines $(V,A,.,q,B,*)$ and $(W,C,.,r,D,*)$ where $(S',X',.,p,Y',*)$ simulates them and each generator of either A or C is a generator of X' and for each generator a of A and c of C and each element x of the free product monoid of A and C $q(xj)*a$ and $r(xj')*c$ are generators of Y'.

Proof: Let $(S',X',.,p,Y',*)$ be a machine which simulates machines $(V,A,.,q,B,*)$ and $(W,C,.,r,D,*)$ with the stated properties. For any generators a of A and c of C and element x of the free product monoid of A and C $q(xj)*a=q(xj)*(aj)=(px*j)k$ is a generator of Y'. Since $(px'*c)k=p(x'j)*(cj)=p(x'j)*e=e$ for any x' of the free product monoid of A and C, $px'*c$ can not be equal to $px*a$ for any x' in the free product monoid.

Conversely, let a and c be generators of X' such that $px*a \neq px'*c$ for all x and x' in the submonoid Y generated by a and c. Each $px*a$ and $px*c$ are generators of Y'. By proposition 4.33, there are nontrivial machines which $(S',X',.,p,Y')$ simulates as a multiprogramming machine.

Let $(V,A,.,q)$ and $(W,C,.,r)$ be the semimachines of these machines and define new output functions on them by $v*a=b=p*a$ for any v in V and $w*c=d=p*c$ for any w in W. If a' is a power of a and c' is a power of c, then $q*(a'a)=(q*a')(qa'*a)=(q*a')(q*a)$ and $r*(c'c)=(r*c')(rc'*c)=(r*c')(r*c)$. Hence, if a' is the m-th power of a and c' is the n-th power of c, then $q*a'$ is the m-th power of b and $r*c'$ is the n-th power of d. Since the submonoid generated by all of the elements of Y' of the form $px*a$ and $px*c$ is a free monoid on the subset of the generators of Y' and the $px*a$ are distinct from the $px*c$ and b and d generate free submonoids, there are homomorphisms k and k' such that $(px*a)k=b$, $(px*c)k=e=(px*c)k'$, and $(px*c)k'=d$. For any x and x' in Y, $(q(x'j))*(xj)=q*(xj)=(p*(xj))k=(p(x'j)*(xj))k$ and $(r(x'j'))*(xj')=r*(xj')=(p*(xj'))k'=(p(x'j')*(xj'))k'$. The machine $(S',X',.,p,Y',*)$ simulates the machines $(V,A,.,q,B,*)$ and $(W,C,.,r,*)$ where B is generated by b and D is generated by d.

There are machines which have $px*a=px'*c$ for some generators a and c and x and x' in the submonoid they generate but can simulate machines with free monoids for both input and output where all outputs from generators of the input monoid are generators of the output monoid. The following example of a machine with two generators for each of the input and output monoids will serve to illustrate this point.

Let the transition and output functions be as follows:

.	t	t'	t''	u	u'	u''
0	t'	t	t	u'	u	u
1	t''	u	t'	u''	t	u''

*	t	t'	t''	u	u'	u''
0	0	0	1	0	1	1
1	1	1	0	1	0	0

The machine $(S',X',.,t,Y',*)$ has $1=t*0=t1*1=t''*1$, but the submachine $(S,X,.,t,Y,*)$ where X is generated by 00, 01, and 10 has the following transition and output functions:

.	t	u
00	t	u
01	u	t
10	t	u

*	t	u
00	00	01
01	01	00
10	11	11

Since $01=t*01\neq t*10=u*10=11\neq u*01=00$, $(S,X,.,t,Y,*)$ simulates machines of the desired type. In particular:

.	q	v
00	q	v
01	v	q

*	q	v
00	00	01
01	01	00

.	r
10	r

*	r
10	11

The homomorphisms are defined such that $(00)j=00$, $(01)j=01$, $(10)j'=10$, $(00)j'=(01)j'=(10)j=e$, $(00)k=00$, $(01)k=01$, $(11)k'=11$, and $(00)k'=(01)k'=(11)k=e$.

Proposition 4.36. Let (S',X',.,Y',*) be an automaton where X' and Y' are both noncyclic free monoids and each s*x such that s is in S and x is one of the generators of X' is one of the generators of Y'. The existence of a state p in S' and of generators a and c of X' such that $px*a \neq px'*c$ for all x and x' in the submonoid generated by a and c is a necessary and sufficient condition for the existence of automata (V,A,.,B,*) and (W,C,.,D,*) where (S',X',.,Y',*) simulates them and each generator of either A or C is a generator of X' and for each v in V and w in W v*a and w*c are generators of Y'.

Proof: Let (S',X',.,Y',*) be an automaton which simulates automata (V,A,.,B,*) and (W,C,.,D,*) with the stated properties. Let q and r be elements of V and W respectively. There is an element p of S' such that (p,q) is in G and (p,r) is in H. The submachine with start state p simulates the submachines with start states q and r. By proposition 4.35, there exist generators a and c such that $px*a \neq px'*c$ for all x and x' in the submonoid generated by a and c.

Conversely, let p, a, and c be such that $px*a \neq px'*c$ for all x and x' in the submonoid generated by a and c. By proposition 4.35, there are machines (V,A,.,q,B,*) and (W,C,.,r,D,*) such that the submachine of (S',X',.,Y',*) with start state p simulates them. The relation G can be defined as the set of all (px,qx) and the relation H as the set of all (px,rx) where x is in the free product of A and C.

The automata of the machines in the previous example, with (t,q) and (t,v) in G and (t,r) and (u,r) in H and the same homomorphisms, illustrates that an automaton might simulate two automata with free semigroups of input and output where each output from any generator of the input is a generator of the output even though it can not simulate any with these same properties where the generators of the monoids of the simulated automata are generators of the monoids of the simulating automaton.

BIBLIOGRAPHY

(A) D. N. Arden. *Delayed Logic and Finite State Machines*, Quarterly Progress Report, R.L.E., M.I.T. 62 (1961) 163-189.

(B) T. L. Booth. *Sequential Machines and Automata Theory*, Wiley (1967).

(BW1) A. R. Bednarek and A. D. Wallace. *Equivalences on Machine State Spaces*, Mathematický Časopis 17 (1967) 3-9.

(BW2) A. R. Bednarek and A. D. Wallace. *Finite Approximants of Compact, Totally Disconnected Machines*, Math. Systems Theory 1 (1967) 209-216.

(CP) A. H. Clifford and G. B. Preston. *The Algebraic Theory of Semigroups*, Amer. Math. Soc. Surveys 7 (1961 and 1967).

(G1) S. Ginsburg. *An Introduction to Mathematical Machine Theory*, Addison-Wesley (1962).

(G2) A. Ginzburg. *Six Lectures on Algebraic Theory of Automata*, Carnegie Institute of Technology (1966).

(G3) A. Ginzburg. *Algebraic Theory of Automata*, Academic Press (1968).

(G4) Y. Give'on. *On Some Properties of the Free Monoids with Application to Automata Theory*, Jour. of Computer and System Sciences 1 (1967) 137-154.

(G5) V. M. Glushkov. *Introduction to Cybernetics*, Academic Press (1966).

(HS) J. Hartmanis and R. E. Stearns. *Algebraic Structure Theory of Sequential Machines*, Prentice-Hall (1966).

(KFA) R. E. Kalman, P. L. Falb, and M. A. Arbib. *Topics in Mathematical System Theory*, McGraw-Hill (1968).

(Ka) T. Kameda. *Generalized Transition Matrix of a Sequential Machine and Its Applications*, Information and Control 12 (1968) 259-275.

(K) S. C. Kleene. *Representation of Events in Nerve Nets and Finite Automata*, Automata Studies, Ann. of Math Studies, 34 (1956) 3-41.

(L) YE. S. Lyapin. *Semigroups*, Translations of Math. Monographs 3 (1963).

(Mc) J. D. McKnight, Jr. *Kleene Quotient Theorems*, Pacific Jour. of Math. 14 (1964) 1343-1352.

(Mi) M. Minsky. *Computation: Finite and Infinite Machines*, Prentice-Hall (1967).

(Mo) E. F. Moore (Ed.). *Sequential Machines* - Selected Papers, Addison-Wesley (1964).

(N) E. M. Norris. *Some Structure Theorems for Topological Machines*, Dissertation, Univ. of Florida, (1969).

(RS) M. O. Rabin and D. Scott. *Finite Automata and their Decision Problems*, IBM Jour. of Res. and Dev. 3 (1959) 114-125.

BIOGRAPHICAL SKETCH

Reverdy Edmond Wright was born 5 August 1933 in Sarasota, Florida. He was graduated from Sarasota High School in June 1951. From September 1951 until January 1956, he attended the Massachusetts Institute of Technology. From June 1956 through May 1958, he served with the U. S. Army. After attending the University of Florida from September 1958 through January 1960, he received with honors the degree of Bachelor of Science in mathematics. From January 1959 through June 1960, he worked as a programmer for the University of Florida Statistical Laboratory. From July 1960 to January 1961, he served with the U. S. Army. From January 1961 to April 1963, he was a programmer for the Statistical Section of the University of Florida's Agricultural Experiment Station. From May 1963 through August 1965, he was the systems supervisor for the University of Florida Computing Center. From September 1965 through August 1969, he was a graduate student in the department of mathematics. During the 1969-70 and 1970-71 academic years, he has been an assistant professor of computer science at the Virginia Polytechnic Institute.

Reverdy Edmond Wright is married to the former Lydia Roeske. They have three children, Tamara, Branwen, and Enid. He is a member of the Association for Computing Machinery, the American Association of University Professors, and the American Association for the Advancement of Science.

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

RG Selfridge

Ralph G. Selfridge, Chairman
Professor of Mathematics

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Alexander R. Bednarek

Alexander R. Bednarek
Professor of Mathematics

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Michael E Thomas

Michael E. Thomas
Professor of Industrial and
Systems Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Kermit N. Sigmon

Kermit N. Sigmon
Assistant Professor of Mathematics

This dissertation was submitted to the Dean of the College of Arts and Sciences and to the Graduate Council, and was accepted as partial fulfillment of the requirements for the degree of Doctor of Philosophy.

June, 1971

Dean, College of Arts and Sciences

Dean, Graduate School

#599 See Main b.e

126

30